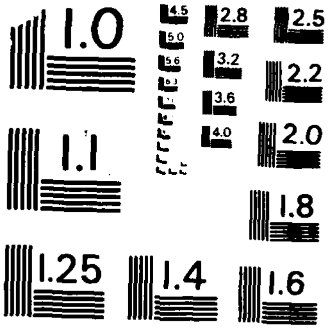
AD-A175 511 SYSTEM LEVEL PROGRAMMING OF THE PRESTON SCIENTIFIC 1/1
ANALOG TO DIGITAL CONV..(U) WESTON OBSERVATORY MA
C J CENTER 29 AUG 86 SCIENTIFIC-4 AFGL-TR-86-0205
UNCLASSIFIED F19628-84-C-0011 F/G 9/2 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

12

AFGL-TR-86- 0205

SYSTEM LEVEL PROGRAMMING OF THE PRESTON SCIENTIFIC ANALOG TO DIGITAL CONVERTER ON THE LSI-11/23 BUS

Christopher J. Center

Weston Observatory
Depaitment of Geology and Geophysics
Boston College
381 Concord Road
Weston, Massachusetts 02193

29 August 1986

Scientific Report No. 4

Approved for Public Release; Distribution Unlimited

Air Foice Geophysics Laboratory
Air Force Systems Command
United States Air Force
Hanscom AFB, Massachusetts 01731

CONTRACTOR REPORTS

This technical report has been reviewed and is approved for publication.

HENRY A. OSSING
Contract Manager

HENRY A. OSSING
Chief, Solid Earth Geophysics Branch

FOR THE COMMANDER

DONALD H. ECKHARDT
Director
Earth Sciences Division

This report has been reviewed by the ESD Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS).

Qualified requestors may obtain additional copies from the Defense Technical Information Center. All others should apply to the National Technical Information Service.

If your address has changed, or if you wish to be removed from the mailing list, or if the addressee is no longer employed by your organization, please notify AFGL/DAA, Hanscom AFB, MA 01731. This will assist us in maintaining a current mailing list.

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b RESTRICTIVE MARKINGS

2a SECURITY CLASSIFICATION AUTHORITY

2b DECLASSIFICATION DOWNGRADING SCHEDULE

3 DISTRIBUTION/AVAILABILITY OF REPORT
APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

4 PERFORMING ORGANIZATION REPORT NUMBER(S)

5 MONITORING ORGANIZATION REPORT NUMBER(S)
AFGL-TR-86-0205

6a NAME OF PERFORMING ORGANIZATION
Weston Observatory
Dept. of Geology & Geophysics

6b OFFICE SYMBOL (If applicable)

7a NAME OF MONITORING ORGANIZATION
Air Force Geophysics Laboratory
Earth Sciences Division

6c ADDRESS (City, State and ZIP Code)
381 Concord Road
Weston, MA 02193

7b ADDRESS (City, State and ZIP Code)
Hanscom AFB, MA 01731
Contract Manager: Henry A. Ossing

8a NAME OF FUNDING/SPONSORING ORGANIZATION
same as Block 7a

8b OFFICE SYMBOL (If applicable)
LWH

9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
F19628-84-C-0011

8c ADDRESS (City, State and ZIP Code)
same as Block 7b

10 SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
62101F	7600	09	AF

11 TITLE (Include Security Classification)
* See block #16

12 PERSONAL AUTHOR(S)
Center, Christopher J.

13a TYPE OF REPORT
Scientific Rpt. #4

13b TIME COVERED
FROM AUG 85 TO AUG 86

14 DATE OF REPORT (Yr., Mo., Day)
86 AUG 29

15 PAGE COUNT
66

16 SUPPLEMENTARY NOTATION
(Block #11*) System Level Programming of the Preston Scientific Analog to Digital Converter on the LSI-11/23 Bus

17 COSATI CODES

FIELD	GROUP	SUB. GR.
08	11	
20	01	

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
LSI-11 Interface, GMAD-4A A/D Converter, System Level Programming, Preston Scientific GMAD-4A, Vibro-Acoustic Measurements, Geokinetic Measurements

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

This report describes the software interface of the Preston Scientific GMAD-4A Analog to Digital Converter with Digital Equipment Corporation's LSI-11/23 host computer. Digital's DRV11-B is used to provide communication between A/D converter and the host.

Tested program examples are written in Macro-11 Version 5.03b Assembly on the RT-11 Version 5.2 operating system. Detailed program comments allow the high level language programmer to understand the sequence of steps necessary for data acquisition on the Preston Scientific A/D Converter.

Note: RT-11 is a Registered Trademark of Digital Equipment Corporation, Maynard, Mass.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. □ DTIC USERS □

21 ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a NAME OF RESPONSIBLE INDIVIDUAL
Henry A. Ossing

22b TELEPHONE NUMBER (Include Area Code)
617-377-3222

22c OFFICE SYMBOL
AFGL/LWH

PREFACE

The author thanks Elizabeth Bergenheim for her initiative, skill and patience in preparing this document for publication. Her meaningful suggestions and overall competence are greatly appreciated.

TABLE OF CONTENTS

1.0 INTRODUCTION

Under contract F19628-84-C0011 with the Air Force Geophysics Laboratory (AFGL), Boston College supports the design, construction, maintenance and operation of a network of state of the art geophysical measurement systems. This report presents system level programming of the analog to digital converter for one element of one such system, the Vibro-Acoustic Measurement System (VAMS) "slave unit" (1). The application of the work to VAMS data acquisition has resulted in a flexible, readily utilized, interactive software package that obtains, controls and processes seismo-acoustic observations. The software includes calibration and recording procedures for sampling 16 channels of data subject to conditions specified by the operator. The Appendix lists a copy of this interactive program written in Digital's Macro-11 Version 5.03b assembly language. The program has been installed and tested on Digital's RT-11 Version 5.2 operating system.

1.1 Functions - For VAMS the GMAD-4A accesses an LSI-11/23 micro computer through a DRV-11B direct memory interface unit that is capable of 250K, 16 bit word transfers per second. The Preston Scientific GMAD-4A is a 16 channel, 15 bit (including sign bit) A/D conversion system. The GMAD-4A has a maximum 16 channel conversion rate of 80 KHz and a full recording scale of plus/minus 10 volts. The system has been supplied with an optional 4K internal FIFO memory buffer.

(1) The AFGL Vibro-Acoustic Measurement System, by H.E. Michel, AIAA PAPER 85-7014, AIAA Shuttle Environment and Operations II Conference, Houston, Texas, November 1985.

To operate the Preston Scientific A/D converter, command words are sent in a Direct Memory Access (DMA) sequence from the host computer. When DMA operations are completed, the programmer may strobe data out of the A/D converter's FIFO memory. Transfer direction, number of words to transfer, and DMA operations are accomplished by the DRV11-B.

System programming examples are presented in a logical, well-documented order. Programmers may follow this sequence of steps to write code in a high level language.

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

2.0 DESCRIPTION OF THE DIGITAL EQUIPMENT CORPORATION DRV11-B

2.1 Introduction - Communication between the A/D converter and the host computer is through the DRV11-B's Word Count Register (WCR), Bus Address Register (BAR), Control Status Register (CSR), and the Data Input (DATI) and Data Output (DATO) Registers. DATI and DATO Registers share the same bus address and their transfer direction is determined by the function bits in the Control Status Register.

The DRV11-B physical address locations are listed in the following table. These addresses are used for in-house testing procedures and may vary from system to system.

ADDRESS	DEVICE REGISTER
772410	Word Count Register
772412	Bus Address Register
772414	Control Status Register
772416	Data Input or Data Output Registers
124	Interrupt Vector location

2.2 Word Count Register - The Word Count Register (WCR) is the word transfer counter. The WCR is loaded with the 2's complement of the number of words to transfer between the host computer and the Preston Scientific A/D Converter. Data Input (DATI) or Data Output (DATO) transfers automatically increment the WCR by one. When the WCR increments to zero, the DRV11-B stops DMA transfers and generates an interrupt through vector location #124.

2.3 Bus Address Register - The Bus Address Register (BAR) contains the first address location of data for DATI or DATO transfers. Data transfers automatically increment the BAR to allow the DMA transfer of data to sequential host memory locations.

2.4 Control Status Register - The Control and Status Register sets communication between the Preston A/D converter and the DRV11-B. The bit information below defines the DRV11-B Control Word. Bit definitions are listed from the Least Significant Bit (LSB) (bit #0) to the Most Significant Bit (MSB) (bit #15)

When applicable, the bit value of High (HI, bit=1) or Low (LO, bit=0) are described.

MSB LSB

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

BIT 0: GO BIT.

The GO bit initializes DMA transfers between the A/D converter and the host computer. This bit is used in conjunction with Cycle Request (CYCREQ) during DATI operations.

BIT 1: FUNCTION 1 BIT.

This bit is known as the FUNCTION 1 (FUNCT1) bit and determines the handshake direction between the DRV11-B and the A/D converter. The inverted value of this bit sets the DRV11-B C1 line (bus direction indicator).

LO: C1 = 1 ; Receive data from the A/D converter.

HI: C1 = Ø ; Send data to the Preston A/D converter.

Note: The C2 line is tied low indicating DMA transfers are in words. Once bit #1 is set LO, DATI transfers will be disabled until the Master Reset (MRESET) bit #2 is sent.

BIT 2: FUNCT2 BIT.

The FUNCTION 2 bit causes a Master Reset (MRESET) on the A/D converter. This bit clears out the current internal status of the A/D converter, clears the Programmed Go (PG), and resets various internal programming codes. This is commonly referred to as the initialize signal.

BIT 3: FUNCT3 BIT. (not available).

The FUNCTION 3 bit is not used by the A/D converter.

BIT 7: READY.

HI: indicates the DRV11-B is ready to accept another command.

LO: indicates the A/D converter is in operation. Sending the "GO" bit HI sends READY LO.

BIT 8: CYCLE.

This bit is used to prime the DMA cycle during DATI operations. For example, to begin data transfers from the host to the A/D converter, send the CYCREQ and GO bits HI in the DRV11-B's CSR.

BIT 11: STAT A. (read only)

This bit is flagged by the Missed Data (MISDAT) line. STAT A is sent HI when the FIFO memory buffer has overflowed. A data buffer overflow will cause the A/D converter to ignore any newly converted data.

3.0 DESCRIPTION OF THE PRESTON A/D CONVERTER CONTROL WORD

3.1 Programming Functions - The Preston Control Word (PCW) determines the programming functions of the A/D converter. PCW is strobed into the A/D converter during the DATI operations. Use of the PCW is demonstrated in examples Number 1 and 2. When applicable, the bit values of HI (bit=1) or LO (bit=0) are described.

MSB															LSB
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BIT 1: CONTROL HANDSHAKE ENABLE.

This bit is hardware set low for the DRV11-B interface.

LO: Used for systems with 1 set of handshake controls. (DRV11-B interface)

HI: Used for systems with 2 sets of handshake controls, one line is input and the other line is output.

BIT 2: SELECT CAM WRITE.

Channel Address Memory (CAM) enables the storage of digitized data in non-sequential order (ex. 0,1,3,2,9) in the A/D converter's First In, First Out (FIFO) memory buffer.

LO: The A/D converter will sequentially store channel data from the First to the Last Channel Address.

HI: Indicates CAM values will be loaded in during DATI operations. The number of CAM values entered will equal the difference between the First and Last Channel Address plus one.

See Example Number 2 for additional help.

BIT 3: SELECT LAST CHANNEL ADDRESS.

LO: The Last Channel Address residing in the A/D converter is used.

HI: The A/D converter will expect a value to be strobed in for the last channel to be digitized.

BIT 4: SELECT FIRST CHANNEL ADDRESS.

LO: The First Channel Address residing in the A/D converter is used.

HI: The A/D converter expects a value to be strobed in for the first channel to be digitized.

BIT 5: SELECT CLOCK DIVISOR.

The Clock Divisor is the channel to channel processing time.

LO: Indicates the A/D converter is to use the default Clock Divisor of 62 (76 octal).

HI: Indicates the Clock Divisor will be strobed in during DATI operations.

The conversion rate for the GMAD-4A is 1 MHz (1000ns. period). The counting frequency is determined by an on-board 5 MHz crystal. The maximum conversion rate per channel is 80 KHz. Please note the following Clock Divisor examples for further detail.

A 80 KHz maximum conversion rate translates into a Clock Divisor having a minimal value of 62.5.

(5 MHz)/(80 KHz) = 62.5 minimum Clock Divisor.

A 80 KHz, maximum conversion rate translates into a 16 channel A/D conversion time of 200 microseconds (usec).

(16 channels)/(80 KHz) = 200 usec.

A Clock Divisor of 70 translates to a conversion rate of 71.42 KHz.

70 usec. = 5 MHz/x

=> x = 71.42 KHz (Note: 71.42 KHz < 80.00 KHz).

BIT 6: RUN CONDITION

Sent in conjunction with the Run Stop Only (RSO) Preston command bit #7 HI.

LO: RSO-STOP. RSO-STOP, Programmed No Go (PNG), causes the A/D converter to wait for a RSO-RUN condition before digitizing after all PCW inputs have been strobed into the A/D converter.

HI: RSO-RUN. This condition, known as Programmed GO (PG), will program the A/D converter to begin digitizing after all the PCW values have been strobed in during DATI operations.

BIT 7: RSO CONDITION

LO: A/D converter will not digitize data until the RSO-RUN bits are sent HI.

HI: The RSO condition is used in conjunction with bit 7 (RUN) to control digitizing of the A/D converter.

BIT 8: SEQUENTIAL or RANDOM MODE.

LO: RANDOM MODE. Enter this condition with bit 7 LO in the PCW. This condition requires the channel addresses for random digitizing. This mode requires the PG condition.

HI: SEQUENTIAL MODE. This condition, entered with bit 7 HI, allows the A/D converter to digitize data sequentially from the First Channel Address to the Last Channel Address.

NOTE: A MRESET command will cause the A/D converter to exit this processing mode.

BIT 9: ENABLE CAM MEMORY.

LO: Enable Channel Address Memory (CAM) programming.

HI: Bypass the CAM for normal random or sequential data digitizing.

BIT 10: BURST MODE.

LO: The clock determines the channel to channel sampling rate.

HI: A group of channels (scans) will be acquired at a predetermined rate set by the Clock Divisor.

BIT 11: EXTERNAL START SOURCE.

LO: The A/D converter will use the internal source to trigger digitizing.

HI: An external source must occur within a given time limit before digitizing is triggered.

BIT 12: CLOCK SOURCE.

LO: The A/D converter's internal clock source will be used.

HI: The A/D converter expects an external clock source to drive the data acquisition process.

BIT 13: REMOTE OR LOCAL PROGRAMMING.

Determines if programming of the Preston A/D converter (Clock Divisor, First and Last Channel Addresses, etc.) will be programmed from the host computer or the front panel.

LO: Program from the front panel.

HI: Program from the host.

BIT 14: LOCAL LOCKOUT.

LO: Programming of the A/D converter may be set through the front panel switches.

HI: The front panel has no effect on operation of the A/D converter. Reset by power down.

BIT 15: SPECIAL MODE BIT.

LO: Required bit state.

HI: Not used.

NOTES:

To selectively process any one channel, set the Last Channel Address equal to the First Channel Address.

4.0 DRV11-B - PRESTON A/D CONVERTER HANDSHAKING (INPUT MODE: DATI)

4.1 DATI Operations - DATI operations enable programming of the A/D converter from the host computer. The 16 bit word instructions are transferred through the DRV11-B parallel line port.

The following instructional steps demonstrate DATI programming of the A/D converter. Examples (Ex.) are included to describe the actual programming sequences involved. For additional help, see Example Number 1.

1. INITIALIZE THE DRV11-B REGISTERS:

Ex. (1) Load the WCR with the 2's complement number of words to transfer.

(2) Load the start of the input array into BAR.

2. ASSURE READY IS HIGH.

READY must be HI to allow DATI operations.

When READY is LO, the A/D converter will not acknowledge a CYCREQ to prime DMA operations.

Ex. Initialize the WCR to zero.

3. SEND A RESET. (optional).

A reset command (all 1's) sent to the A/D converter will perform the following functions:

(1) Return the system to input control word mode. A new control word must be entered to start the A/D converter. This could be the RSO-RUN condition.

(2) Leave the previously programmed A/D converter unchanged.

(3) Clear the FIFO memory buffer in the A/D converter.

Ex. Load a RESET into the first array location to be strobed into the A/D converter.

4. SEND THE PCW

Send the PCW followed by the PCW values that reside in an array for DMA transfers.

Ex. Load the PCW into the second array location.

5. SEND PCW VALUES.

The next values sent to the A/D converter correspond to the HI bits entered in the PCW in the order of MSB to LSB.

Ex. Load the next sequence of input array elements with the Clock Divisor, First and Last Channel Address.

6. SEND THE DRV11-B RSO CONDITION.

The RSO condition sent determines how the A/D converter will begin processing. A RSO-RUN condition will send the A/D converter into digitizing mode after CYCREQ and GO have been strobed.

Ex. Load the last element of the input array with the desired RSO condition.

7. SEND FUNCT1 BIT.

Sending FUNCT1 HI sets DATI transfers.

Ex. Load the DRV11-B CSR with the FUNCT1 bit HI.

8. SEND FUNCT2 BIT.

FUNCT2 sent HI resets the PG and various lines of communication within the A/D converter. The FUNCT1 bit remains high during this stage.

Ex. (1) Load the DRV11-B CSR with the FUNCT2 bit HI.

(2) Clear the DRV11-B CSR FUNCT2 bit.

9. WAIT.

A delay time of 100 microseconds allows the A/D converter to set up the necessary program control before digitizing data. (Digitizing begins when the GO-CYCREQ condition is sent).

10. SEND CYCLE AND GO CONDITIONS.

The GO bit sends READY LO and allows DMA operations.

CYCLE is sent to prime the DMA transfer. The FUNCT1 bit remains HI to maintain DATI operations.

Ex. Load the DRV11-B GO and CYCREQ bits.

After the CYCREQ has been primed, the interface will enter DMA operations under hardware control until the WCR increments to zero. The following steps describe hardware functions.

1. THE A/D CONVERTER GENERATES A CYCREQ.

When CYCREQ is asserted, input data is sent to the DATO register. Control bits are latched into the DRV11-B DMA control, and the DRV11-B sends BUSY LO.

Preston's CYCREQ allows access to the LSI-11 bus.

2. DRV11-B GENERATES A BUSY.

At the end of a cycle, the DRV11-B causes the WCR and BAR to be incremented. BUSY goes HI while READY remains LO. BUSY going active (active LO) causes the A/D converter to negate CYCREQ and the trailing edge of BUSY enters the indata word into the A/D converter.

3. THE A/D CONVERTER GENERATES A CYCREQ.

The A/D converter generates another CYCREQ. The user is only required to prime the first DMA cycle.

4. REPEAT ANOTHER CYCLE.

The cycle of CYCREQ and BUSY continues until the WCR is incremented to zero.

5. FINISHED: WCR=0.

When data transfers are complete, the DRV11-B will send READY HI, and the WCR will have been incremented to zero.

The DRV11-B at this time will generate an interrupt.

The DRV11-B Control/Status word will now read #202 (octal).

NOTE: If BURST MODE is selected (SINGLE CYCLE LO), only one CYCREQ is needed for the complete transfer of the specified number of words. The DRV11-B does not wait for a CYCREQ to return from the A/D converter but continually strobes data in or out of memory.

4.2 Programming Functions - Programming the Preston A/D converter is accomplished by sending the Preston Control Word (PCW) during the DATI mode. Send the high bits of those values being sent to the A/D converter. Starting with the most significant bit of the PCW, enter the A/D converter programming function values for each PCW bit entered HI. For example, to strobe in a PCW of 20470 octal (as shown in Example No. 1, the following data strobes would contain the Clock Divisor, First Channel Address, and Last Channel Address.

4.3 Example Number 1: Initialization - Example Number 1 initializes the Preston Scientific A/D converter using the programming capabilities of the PCW.

```
     .TITLE INPUT
;*************************************************************************;
;*                        EXAMPLE NUMBER 1                               *;
;*  Macro-11 Assembly language example of programming the Preston Scientific  *;
;*  A/D converter.                                                       *;
;*                                                                       *;
;*  NOTE: This example was written and tested on a LSI-11/23 16 bit micro-  *;
;*        computer.  After execution of this program, data may be strobed  *;
;*        out of the A/D converter's FIFO memory buffer.                 *;
;*                                                                       *;
;*************************************************************************;
; DEFINE SYSTEM MACROS
;-
        .MCALL  .EXIT
;+
; ASSIGN INTEGER*2 MEMORY LOCATIONS.
;-
MEMORY: .WORD   177777   ; MEMORY (1) = RESET, (octal).
        .WORD   20470    ; MEMORY (2) = PCW.
        .WORD   77       ; MEMORY (3) = Clock Divisor.
        .WORD   0        ; MEMORY (4) = First Channel Address
        .WORD   15.      ; MEMORY (5) = Last Channel Address.  (15 decimal)
        .WORD   300      ; MEMORY (6) = RSO-RUN condition (octal).
;+
; LOAD THE PRESTON A/D REGISTERS AND INITIATE A RSO-RUN CONDITION.
;-
INPUT:  MOV     #0,@#172410       ; Assure RDY is HI by setting the WCR to zero.
        MOV     #-6,@#172410      ; Load WCR with No. of words to transfer.
        MOV     #MEMORY,@#172412  ; Load BAR with the start address for DMA
                                    transfer
        MOV     #6,@#172414       ; Send FUNCT1, FUNCT2 HI.
        MOV     #2,@#172414       ; Send FUNCT2 LO, maintain FUNCT1 HI.
        JSR     PC,DELAY          ; Call the 100 usec delay subroutine.
XXX:    MOV     #403,@#172414     ; Send GO bit, cause RDY to go LO
                                  ;   Maintain FUNCTI HI and assure CYCREQ
                                  ;     is HI (BUSY LO).
        .EXIT                     ; Exit program
;+
; TIME DELAY SUBROUTINE...
;-
DELAY:  MOV     R0,-(SP)          ; SAVREG...
        MOV     #100.,R0          ; Set R0 with a counter.
        SOB     R0,.              ; Wait..
        MOV     (SP)+,R0          ; RESREG...
        RTS     PC                ;   and return.
;
        .END    INPUT             ; End of program.
;*************************************************************************;
```

Since an RSO-RUN condition has been entered during control word input, the A/D converter is now digitizing. Data will be stored in the FIFO memory buffer at a rate predetermined by the Clock Divisor. The program will start converting data when the last instruction has been strobed into the A/D converter (indicated by "XXX:" in the above program).

The following chart describes the PCW entered in Example Number 1.

BITS	DESCRIPTION
5,4,3	Indicates the Clock Divisor, First and Last Channel values will be entered.
8	Indicates sequential processing of multiplexer channels.
13	Indicates remote (computer) programming mode.

4.4 Example Number 2: Use of Channel Address Memory - Example Number 2 indicates use of Channel Address Memory on the A/D converter. CAM allows data acquisition of channels in a non-sequential order.

```
            .TITLE     CAMTST
;****************************************************************************;
;*                              EXAMPLE NUMBER 2                             *;
;*  CAM processing demonstration.  Sequence thru 4 channels in the order of   *;
;*  4,2,1,0.                                                                 *;
;****************************************************************************;
        .MCALL  .EXIT   ; Define library macros.
;+
; ASSIGN INTEGER*2 MEMORY LOCATIONS.
;-
MEMORY:  .WORD   177777  ; MEMORY (1) = RESET. (octal).
         .WORD   21474   ; MEMORY (2) = PCW
         .WORD   77      ; MEMORY (3) = Clock Divisor.
         .WORD   0       ; MEMORY (4) = First Channel Address.
         .WORD   3       ; MEMORY (5) = Last Channel Address.
;+
; CAM VALUES FOR CHANNEL SEQUENCING.
;           .WORD   4       ; MEMORY (6) = Save channel no. 4 first.
            .WORD   2       ; MEMORY (7) = Save channel no. 2.
            .WORD   1       ; MEMORY (8) = Save channel no. 1.
            .WORD   0       ; MEMORY (9) = Save channel no. 0 last.
            .WORD   300     ; MEMORY (10)= RSO-RUN condition.
;+
; PROGRAM THE PRESTON A/D.
;-
CAMTST:     MOV     #0,@#172410        ; Assure READY is HI, set the WCR to zero.
            MOV     #-12,@#172410      ; Load WCR with no. of words to transfer.
            MOV     #MEMORY,@#172412   ; Load BAR with the first DMA address.
            MOV     #6,@#172414        ; Send FUNCT1, FUNCT2 HI.
            MOV     #2@#172414         ; Send FUNCT2 LOW, maintain FUNCT1 HI.
            JSR     PC, DELAY          ; Wait 100 usec.
            MOV     #403@#172414       ; Send GO bit, causes RDY to go LO.
                                       ; Note that this command maintains FUNCT1 HI
                                       ;  and send CYCREQ HI to prime DMA transfer.
            .EXIT                      ; End of program
;+
; DATA IS NOW IN THE PRESTON A/D FIFO MEMORY BUFFER WAITING TO BE STROBED INTO
; HOST MEMORY.
;-
DELAY:      MOV     RO,-(SP)
            MOV     #100.,RO
            SOB     RO,.
            MOV     (SP)+,RO
            RTS     PC
;
            .END    CAMTST
;****************************************************************************;
```

The Preston A/D is now in A/D conversion, digitizing thru channels 4, 2, 1, and 0

PRESTON A/D CONVERTER -- DRV11-B HANDSHAKING (OUTPUT MODE: DATO)

DATO mode operations are similar to DATI mode operations. In DATI mode, the host program asserts the first Cycle Request (CYCREQ) initiating DMA transfers. In DATO mode, the A/D converter internally asserts the first CYCREQ.

After execution of Example Number 1 or 2, the A/D converter is in the RSO-RUN condition and storing digitized data in the A/D converter's FIFO memory buffer. The next logical steps are to transfer data from the FIFO memory buffer to host memory.

The following steps with Macro-11 examples describe the DATO programming.

1. ASSURE THE WCR IS ZERO.

   Upon completion of the DATI processing steps, the WCR will have been incremented to zero.

   Ex. Wait for the WCR to be incremented to zero from DATI operations.

2. LOAD THE WCR.

   To strobe 16 channels from the A/D converter FIFO memory, load the WCR with the 2's complement of the number of channels to sample.

   Ex. Load the "negative number of channels" into the WCR.

3. ASSIGN A STORAGE ARRAY FOR DMA OUTPUT.

   Assign an array (sequential memory locations) for digitized output from the A/D converter. (See Example Number 3).

   Ex. Load the BAR with the number of channels to read..

4. SEND GO BIT.

   Upon completion of steps 1, 2 and 3, the user sends the GO bit to prime DMA operations. In DATO mode, the A/D converter will send the first CYCLE to prime DMA operations.

   Ex. Load the DRV11-B GO bit.

The following steps are hardware generated.

1. The A/D converter generates a CYCREQ when data is available.

2. DRV11-B's response is to generate a BUSY.

   Similar to DATI, BUSY negates CYCREQ and reads in the data on the cards input bus.

3. BUSY is removed after DRV11-B reads the data on the line.

   This trailing edge will request the next data word from the A/D converter if a FIFO memory buffer exists on the system.

4. The A/D converter generates another CYCREQ when data becomes available.

5. This cycle of CYCREQ and BUSY (steps 6-9) continues until WCR = 0.

6. When WCR = 0, READY is sent HI by the DRV11-B. (Completion).

   The DRV11-B Control/Status Register should now contain #200.

4.5 Example Number 3: Example of A/D DATO Programming Steps - Example Number 3 demonstrates how to strobe data out of the A/D converter FIFO memory buffer after DATI completion. Before data is strobed out, a time delay of 2.025 msec must be issued before further processing. The delay time may be broken down according to the following table:

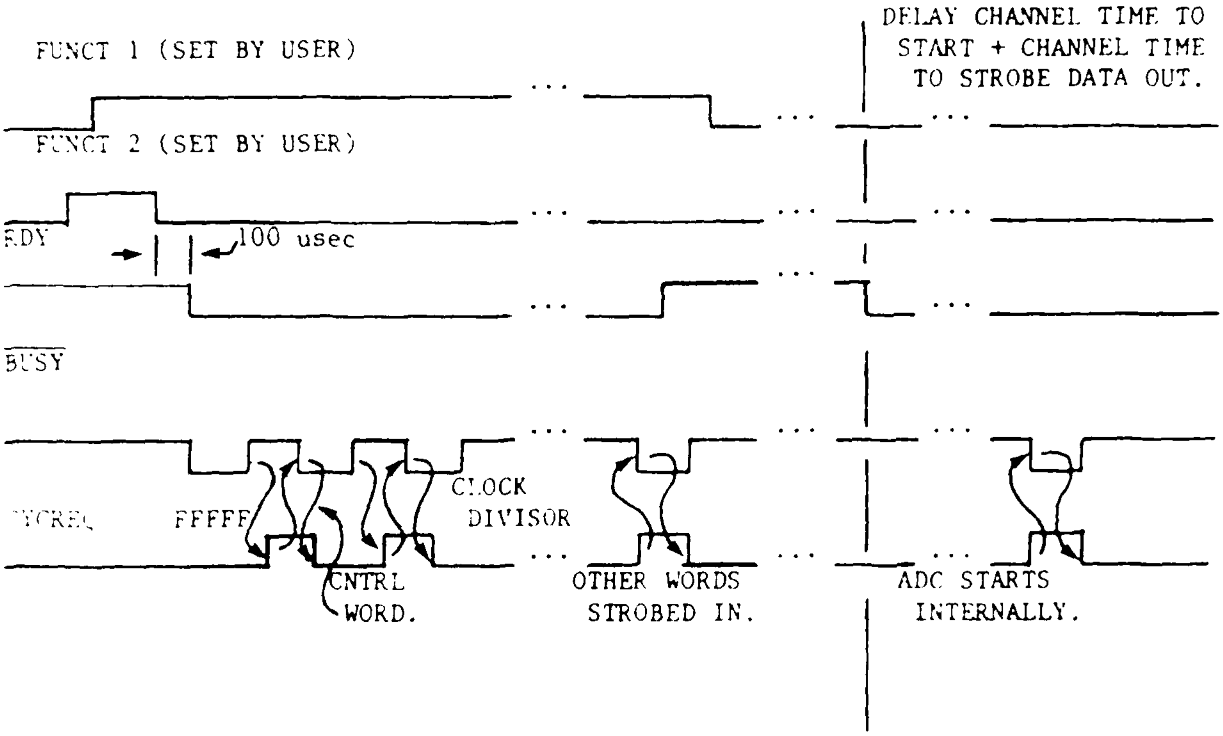
1.7 msec.	Time for RSO-RUN acknowledge.
0.2 msec.	Time for 16 channel conversion.
0.125 msec.	Extra channel delay time.

```
        .TITLE      OUTPUT
;*************************************************************************;
;*                           EXAMPLE NUMBER 3.                           *;
;*  DEMONSTRATION OF DATA TRANSFER FROM THE PRESTON A/D FIFO MEMORY TO THE    *;
;*  HOST COMPUTER.                                                       *;
;*************************************************************************;
;   INITIALIZE SYSTEM MACROS...
        .MCALL  .EXIT
;+
;  INITIALIZE AN ARRAY FOR PRESTON A/D OUTPUT.
;-
MEMORY:  .BLKW   16.      ; Define the data array "MEMORY(16)",
;+
; DATA TRANSFER
;-
OUTPUT: MOV     #16.,@#172410      ; Load the WCR with the 2's complement
        NEG     @#172410           ;   No. of channels to read.
;       MOV     #MEMORY,@#172412   ; Load the BAR with the start address
        MOV     #40000,@#172412    ; Load the BAR with the start address for DMA.
        BIC     #2,@#172414        ; Assure FUNCT1 is LO.
        JSR     PC,DELAY           ; Call the 100 ms. delay subroutine.
        BIS     #1,@#172414        ; Initialize DATO DMA transfer. (Send GO).
        .EXIT                      ; Exit program.
;+
; TIME DELAY SUBROUTINE....
;-
DELAY   MOV     R0,-(SP)           ; Save Registers (SAVREG)
        MOV     #100.,R0           ; Set R0 with a counter.
        SOB     R0,.               ; Wait..
        MOV     (SP)+,R0           ; Restore Registers(RESREG)
        RTS     PC                 ;   and return.
;
        .END    OUTPUT
;*************************************************************************;
```

The DRV11-B will now transfer 16 channels of data to Host memory. The WCR incremented to zero will indicate completion of this routine.

4.6 Example Number 4: Bit Status of the FUNCT1

```
;*************************************************************************************;
;*                              EXAMPLE NUMBER 4                                     *;
;*  DISPLAYS THE BIT STATES OF THE FUNCT1, FUNCT2, RDY, BUSY and CYCREQ              *;
;*  during programming and operation of the A/D converter.                           *;
;*                                                                                   *;
;*************************************************************************************;
```



5.0 REFERENCES

GM Series Analog-To-Digital Conversion Systems Interface Manual, 1985, Preston Scientific, Anaheim, California.

Microcomputer Handbook, 1977, 1978, Digital Equipment Corporation, Maynard, Massachusetts.

RT-11 Version 5 Programmer's Refeience Manual, Volume 3A, Macro-11 Language Reference Manual, July 1984, Digital Equipment Corporation, Maynard, Massachusetts.

NOTE: Example Numbers 1 through 4 designed by Christopher J. Center, Weston Observatory, 22 Aug 86.

6.0 ACRONYMS/ABBREVIATIONS

A/D	Analog to Digital.
BAR	Bus Address Resister (DRV11-B).
BURST MODE	Preston A/D converter command. (PCW bit #10 HI).
BUSY	BUSY HI enables DMA tranfers. Negates CYCREQ.
CAM	Channel Address Memory. (A/D command).
CSR	Control Status Register
CYCREQ	Cycle Request. Asserted by the A/D converter to gain access of the LSI-11/23 bus.
DATI	Data Input Register (memory to DRV11-B transfer).
DATO	Data Output Register. (DRV11-B to memory transfer).
DMA	Direct Memory Access.
FIFO	First In, First Out.
FUNC1	Determines A/D converter - DRV11-B handshake direction. (DRV11-B CSR bit #1).
FUNC2	A/D converter initialization. (DRV11-B CSR bit #2).
GO	Enables DMA transfers. Sends READY LO. (DRV11-B CSR bit #0)
HI	HIGH (bit = 1).
LO	LOW (bit = 0).
LSB	Least Significant Bit. (bit = 0).
MISDAT	Missed Data. Indicates the Preston A/D converter's FIFO memory buffer is full.
ms.	Milliseconds (1 sec. = 100 ms.).
MSB	Most Significant Bit. (bit #15).
MRESET	Master Reset. Word with all bits HI. (177777 octal).
PCW	Preston Command Word. Responsible for programming of the A/D converter.
PG	Programmed Go.
PNG	Programmed No Go.

RDY READY.

READY Indicates the DRV11-B may accept another command. (RO). (DRV11-B Control/Status bit #7).

RO Read only.

RSO Run Stop Only. (PCW bit #6).

RSO-RUN Start A/D conversion. (PCW bit #6 HI and bit #7 HI).

RSO-STOP Stop A/D conversion. (PCW bit #6 HI and bit #7 LO).

SINGLE CYCLE A/D converter line. LO when BURST MODE is selected.

STAT A Status command sent from the A/D converter. This bit beomes HI when the A/D converter's FIFO memory buffer is full. (DRV11-B CSR bit #11).

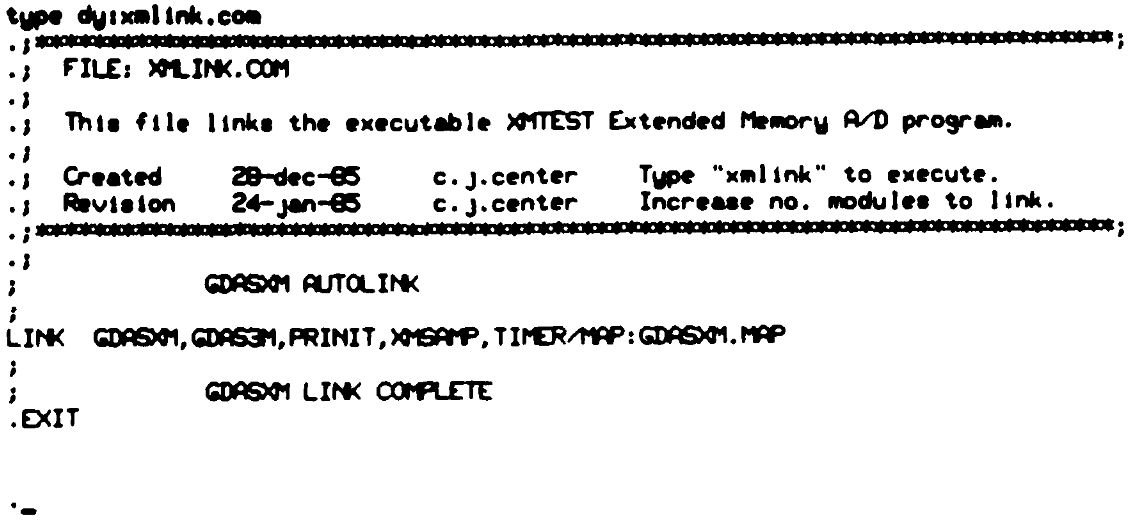
usec. Microseconds (1 sec. = 1000 usec.).

VAMS Vibro-Acoustic Measurement System.

WCR Word Count Register. (DRV11-B).

6.1 MACRO-11 ASSEMBLER LIST.

BIC	Bit Clear. Clear the matching (octal) bits.
BIS	Bit Set (Logical OR operation). Set the matching (octal) bits.
.BLKW	Reserve memory for 16 bits or multiples of 16 bits.
CAMTST	Program name.
.EXIT	End of program.
INPUT	Program name.
JSR	Jump to subroutine.
LOOP	Loop variable. (Similar for the Fortran continue statement).
MACROS	Macro-11 subroutines.
.MCALL	Allow macro library access.
MOV	Move Instruction.
NEG	Negate.
OUTPUT	Program name.
RTS	Return from subroutine.
SOB	Subtract from the first value and branch to the second value if not result no zero.
WAIT	Loop variable. (Similar for the Fortran continue statement).
.WORD	Load memory with a 16 bits value.



```
type dy:xmlink.com
.;***************************************************************************;
.;  FILE: XMLINK.COM
.;
.;  This file links the executable XMTEST Extended Memory A/D program.
.;
.;  Created     28-dec-85     c.j.center    Type "xmlink" to execute.
.;  Revision    24-jan-85     c.j.center    Increase no. modules to link.
.;***************************************************************************;
.;
;               GDASXM AUTOLINK
;
LINK  GDASXM,GDAS3M,PRINIT,XMSAMP,TIMER/MAP:GDASXM.MAP
;
;               GDASXM LINK COMPLETE
.EXIT


._
```

APPENDIX

```
type dy:gdasxm.mac
type dy:gdasxm.mac

!hhco9
!hco9

.       .TITLE  GDASXM
        .IDENT  /V02.20/
        .SBTTL  EDIT LOG
;*****************************************************************************;
;*                           WESTON OBSERVATORY                              *;
;*                      DEPT. OF GEOLOGY AND GEOPHYSICS                      *;
;*               GEOPHYSICAL DATA ACQUISITION SYSTEM SOFTWARE                *;
;*  FILE:  GDASXM.MAC                                                        *;
;*                                                                           *;
;*  CREATED   22-MAY-86   C.J.CENTER   Enhanced programming techniques to    *;
;*                                     allow common subroutines between all  *;
;*                                     sampling programs.                    *;
;*  REVISON   10-JUN-86   C.J.CENTER   Install optional D/A calibration.     *;
;*****************************************************************************;
;*  LINK FILES:                                                              *;
;*      1.  GDAS3M.MAC:   Subroutine file.                                   *;
;*      2.  PRINIT.MAC:   Preston GMAD-4A converter initialization.          *;
;*      3.  XMSAMP.MAC:   RT11-XM data sample handler.                       *;
;*  INCLUDE FILES:                                                           *;
;*      1.  GDAS1I.INC:   Device equivalences and "IF" conditionals.         *;
;*      2.  GDAS2I.INC:   Globals.                                           *;
;*      3.  GDAS3I.INC:   Data definitions.                                  *;
;*      4.  GDAS4I.INC:   Macros.                                            *;
;*****************************************************************************;
;*                        EXTENDED MEMORY SAMPLING                           *;
;** RECORD NO.1                                                              *;
;*                                                                           *;
;*    BLOCK NO. 0...                                                         *;
;*          BYTE 0:       Blank.                                             *;
;*          BYTE 1-511.: User comments.                                      *;
;*    BLOCK NO. 1...                                                         *;
;*      This block contains sampling header information:                     *;
;*          WORD 1:   DOUBLE PRECISION (HIGH ORDER) NUMBER OF SCANS SAMPLED.*;
;*          WORD 2:   DP (LOW ORDER) NUMBER OF SCANS SAMPLED.                *;
;*          WORD 3:   ERC CLOCK FREQUENCY. (number of interrupts /sec).      *;
;*          WORD 4:   SAMPLE RATE (samples/sec.)                             *;
;*          WORD 5:   BIT MAP FOR SAMPLED CHANNELS (0-15)                    *;
;*          WORD 6:   ERC START TIME:   Days & hours.                        *;
;*          WORD 7:                     Minutes & seconds.                   *;
;*          WORD 8:   ZERO.                                                  *;
;*          WORD 9:   D/A CALIBRATION FLAG.   (0=OFF.  1=ON.)                *;
;*                                                                           *;
;** RECORD NO.2 - END OF FILE.                                               *;
;*      These records contain scan information recorded from the Preston    *;
;*      A/D converter or the ERC clock composed of the following format:    *;
;*          WORD 1:   START OF SCAN INDICATOR.  (072525 OCTAL)               *;
;*          WORD 2:   DIVIDED INTO THE FOLLOWING 2 BYTES.                    *;
;*                      BYTE 1:    ERC FLAG.                                  *;
;*                                   1, ERC CLOCK DATA IN NEXT TWO WORDS.     *;
;*                                   0, CHANNEL DATA FOLLOWS.                 *;
;*                      BYTE 2:    NUMBER OF DATA WORDS IN SCAN.              *;
;** OUTPUT SCAN EXAMPLE.                                                      *;
;*          072525 010000 177747 000144 177750 000005 177664 000101           *;
;*          177750 000104 177704 000074 177726 000071 177722 000056           *;
;*          177665 000043 072525 001001 000005 013507                         *;
;**************************************************************************;
        .MCALL      .CSIGEN,.PRINT,.WRITW,.CLOSE,.EXIT,.SCCA,.GTLIN
        .MCALL      .RDBBK,.WDBBK
        .INCLUDE    \GDAS1I.INC\
        .INCLUDE    \GDAS2I.INC\
        .GLOBL       DASTRT,DASTOP,INTON,INTOFF,CLKTIM
        .INCLUDE    \GDAS3I.INC\
;+
    .SBTTL  PROGRAM CRASH ADDRESS RECOVER.
;-
LC=.                                ; Location 4,6 handle illegal address ref.
.=4+LC                              ; Location 10,12 handle ill. instruction.
        .WORD   6,0,12,0            ; ERR handling definitions for the CP.
;+
    .SBTTL  ACQUIRE AND SET SAMPLING PROCEDURES.
;-
        .PSECT  PROG
BEGIN:  .PRINT  #HELLO              ; Display program title to user.
        JSR     PC,DASTRT           ; Query user for D/A calibration option.
        JSR     PC,ADCOMM           ; Query user for data sampling parameters.
        JSR     PC,BITMAP           ; Generate a bitmap for channels being sampled
        JSR     PC,PRINIT           ; Initialize the PRESTON A/D converter.
        JSR     PC,INTKIL           ; Kill undesireable interrupts.
;+
    .SBTTL  DISPLAY SAMPLING PARAMETERS...
;-
        .PRINT  #INPUT              ; Redisplay user inputs as a check....
        JSR     PC,TYSAMP           ;   sample rate,
        JSR     PC,TYSCAN           ;   no. of scans,
        JSR     PC,TYCMAP           ;   and channels being sampled.
        JSR     PC,GETCOM           ; Query user for comments.
;+
    .SBTTL  QUERY USER FOR OUTPUT DATA FILE.
;-
        .PRINT  #ENTER                  ;PRINT INFO FOR CSIGEN FILENAMES
        MOV     SP,SPTEMP               ;SAVE SYSTEM STACK POINTER
        .CSIGEN #DEVHND,#DEFEXT,#0 ;AND OPEN FILE(S)
        MOV     SPTEMP,SP               ;RESTORE STACK PTR(IGNORE CSIGEN OPTIONS)
;+
    .SBTTL  WRITE COMMENTS TO OUTPUT FILE,  INIT OUTPUT BUFFER.
;-
        JSR     PC,VIRWIN               ; Initialize virtual memory.
        MOV     WINDOW+W.NBAS,OUTBUF ; Initialize the output buffer.
        JSR     PC,WRTCOM               ; Write comments to output file.
        JSR     PC,WRTHED               ; Write header to buffer. (Assign ptrs).
```

```
;+
        .SBTTL  INITIALIZE I/O DEFINITIONS.
;-
        CLR     SCANS               ; "SCANS" is a counter for XM...
        MOVB    #1,TRIGON           ; Init. sample-completion trigger to OFF.
;+
        .SBTTL  REQUIRE A <CR> BEFORE SAMPLING BEGINS.
;-
        CLR     R0                  ; Clear register and...
        .GTLIN  R0,#OKSAMP          ;   wait for a <cr> before sampling...
        .PRINT  #TRIGOU             ; Write a sample message...
        CLR     SCCAST              ; Clear .SCCA status word...
        .SCCA   #CAREA,#SCCAST      ;   and disable ^C.
;+
        .SBTTL  START INTERRUPTS AND SAMPLE.
;-
        JSR     PC,CLKTIM           ; Get the current time.
        JSR     PC,INTON            ; Set interrupts...
SIT:    TSTB    TRIGON              ; Wait....
        BLT     DONE                ;   and exit when complete.
        TSTB    SAMPFL              ; Time to sample ??
        BEQ     20$                 ;   NO - continue
        JSR     R5,XMSAMP           ;   YES- jump to sample subroutine...
        CLRB    SAMPFL              ; Reset sample flag.
20$:    TSTB    ERCFLG              ; Time to read the clock ??
        BEQ     SIT                 ;   NO - continue.
        JSR     R5,XERCNT           ;   YES- jump to record time.
        CLRB    ERCFLG              ; Reset the ERC time flag.
        JMP     SIT                 ; Wait for more interrupts...
;
DONE:   MTPS    PR7                 ; Restore system priority
        JSR     PC,INTOFF           ; Shut off interrupts...
        CLR     SCCAST              ; CLEAR .SCCA STATUS WORD
        .SCCA   #CAREA,#0           ; ENABLE ^C.
        JSR     PC,VIRWRT           ; Write A/D data to output file
;+
        .SBTTL  DISPLAY FINAL SAMPLING MESSAGES.
;-
        MOV     #AGOTIM,R2          ;WHERE TO PUT ASCII TIME OF TRIGGER
        JSR     R5,ASCTIM           ;CONVERT ERC TIME TO ASCII
        .WORD   GOTIME              ;ADDRESS OF ERC TIME OF TRIGGER

        MOV     #AENTIM,R2          ;WHERE TO PUT ASCII END TIME
        JSR     R5,ASCTIM           ;CONVERT END TIME
        .WORD   ERCMSB              ;END TIME IS LAST ERC READOUT

        .PRINT  #TTRIG              ;PRINT TIME OF TRIGGER AND END TIME
        CMP     SCANS,NSCAN
        BEQ     1$                  ;BRANCH WHEN ALL SCANS ACQUIRED
        .PRINT  #OFLOW              ;MUST HAVE OVERFLOWED
;
1$:     MOV     SCANS,R1            ;# OF SCANS ACQUIRED
        JSR     R0,OTOAD            ;CONVERT TO ASCII DECIMAL
        .WORD   TSCANS          ;PUT ASCII THERE
        .PRINT  #SCANM2         ; Indicate to the user the
        .PRINT  #TSCANS         ;    # of scans recorded.
;+
    .SBTTL  END OF RT11-FB SAMPLING
;-
        TSTB    DKCAL           ; Is calibration in effect ??
        BEQ     999$            ;   NO - normal exit...
        JSR     PC,DASTOP       ;   YES- clear D/A values.
999$:   .EXIT                   ; Exit.
        .END    BEGIN
```

```
type dy:gdas11.inc
!hco2

;***************************************************************************;
;*  INCLUDE FILE:  GDAS11.INC                                              *;
;*                                                                         *;
;*  CREATED   21-MAY-86   C.J.CENTER   GDAS SOFTWARE EQUIVALENCE FILE.     *;
;***************************************************************************;
;+
;   ***   SAMPLE DEVICE COMPILE FLAGS   ***
;         NOTE:  YOU "MUST" CHOOSE ONLY "ONE" OF THE FOLLOWING CONDITIONALS
;                TO COMPILE EACH SAMPLING PROGRAM.       ( 0=OFF, 1=ON )
;-
FSAMP = 0       ;  Foreground / Background sampling.
XSAMP = 1       ;  Extended Memory sampling.
RSAMP = 0       ;  Bubble, Floppy Disk, Hard disk sampling.
TSAMP = 0       ;  Kennedy tape sampling.
;
CLKVEC = 100    ; Clock vector location.
DRVVEC = 124    ; DRV11B Interrupt vector location.
ERRWRD = 52     ; ADDR OF ERROR TYPES FOR PROGRAMMED REQUESTS
;
ERCCSR = 167770 ; ERC Control/Status Register.
ERCOUT = 167772 ; ERC Output Register.
ERCIN  = 167774 ; ERC Input Register.
ERCVEC = 170    ; ERC vector location.
;
DAC1 = 176750   ; D/A Converter line #1 address.
DAC2 = 176752   ; D/A Converter line #2.
;
        .IF  NE  RSAMP
BUBCSR = 177150 ; Bubble Memory Control/Status Register.
BUBVEC = 270    ; Bubble memory vector location.
BUBPRI = 272    ;                  priority loc.
        .ENDC
;+
;   END DATA EQUIVALENCES.
;-

.
```

```
type dy:gdas2i.inc
!hco2

;****************************************************************************;
;*  INCLUDE FILE:  GDAS2I.INC                                               *;
;*                                                                          *;
;*  CREATED   21-MAY-86   C.J.CENTER   GDAS sofware file containing globals *;
;*            30-MAY-86                Install Kennedy Tape globals.        *;
;****************************************************************************;
; pointers and tables...
        .GLOBL  OUTPTR,OUTBUF,OUTEND,BLKNUM,TIMPTR
        .GLOBL  ARGBLK,DAY,HOUR,MINUTE,SECOND

; ERR codes...
        .GLOBL  EROPEN,EREOF,ERCBUF,ERHARD,WRERR,ERRWRD,ERRMEM,IOFATL

; misc storage...
        .GLOBL  BLKNUM,ERCLSB,ERCMSB,SCCAST,CHMAP1
        .GLOBL  AREA,CAREA,TRIGOU,NOROOM,SCANS,CLKVEC,BASRAT,PR0,PR7
        .GLOBL  CHANM1,SCANM1,OKSAMP,INPUT,CHANM2,ARATE,PLUS

; comments...
        .GLOBL  WRTCOM,GETCOM,COMBUF,CBUFEN,COMEND,LINBUF,CPROMP
        .GLOBL  ARATE,ACHSAM,TSCANS,COMMNT,CINFO,CHLEFT,COMOVF
        .GLOBL  OTOAD,OTOAD1,PUTOCT,GETDEC,GETOCT,WRTHED
        .GLOBL  GETCLK,GETSAM,GETCHN,GETSCN,TYSAMP,TYCMAP,TYSCAN,ASCTIM,BITMAP

; misc storage for I/O inputs...
        .GLOBL  NCHAN,NSCAN,SAMRAT,CLKRAT,DKCAL,DAC.M1,DAC.M2,DAC.M3
        .GLOBL  ERCINT,DAC1,DAC2,PULSE

; sample interrupt globals....
        .GLOBL  SAMPLE,SCANID,OFLOW,TRIGON,GOTIME,SAMBUF
        .GLOBL  ADCOMM,PRINIT,ERCIN,ERCINT

; FB - specific globals...
        .IF  NE  FSAMP
          .GLOBL  ENTER,ENDPAD,DEVHND,DEFEXT,SPTEMP
          .GLOBL  PROFIT,WRTDAT,FBEND
        .ENDC

; XM - specific globals....
        .IF  NE  XSAMP
          .GLOBL  ENTER,ENDPAD,DEVHND,DEFEXT,SPTEMP,GOTIME,SAMBUF
          .GLOBL  W.NOFF,W.NBAS,SAMPFL,WNDCNT,WINDOU
          .GLOBL  W.NRID,R.GID,INOFF,SAMPLE
          .GLOBL  XMBUF,XAREA,XERROR,EROFLG,XERCNT,SAMPFL,INTOFF
          .GLOBL  XMSAMP,VIRWIN,VIRWRT,ERR,ERRNO
        .ENDC

; RT - specific globals....
        .IF  NE  RSAMP
          .GLOBL  ENTER,ENDPAD.DEVHND,DEFEXT,SPTEMP
          .GLOBL  BLKADD,BLKEND, BLKN0,BLKN1,BLKFLG
          .GLOBL  QAREA,FBEND,PROFIT,LSTWRT,SETBLK
        .ENDC
```

```
; TAPE - specific globals....
        .IF  NE  TSAMP
          .GLOBL  RECBUF,RECADD,RECEND, RECN0,RECN1,RECFLG,RECNUM
          .GLOBL  RWTAPE,WTAPE,ENTAPE,QUERY,KENCSR,KENOUT,KENIN
          .GLOBL  SETREC,TYSCN2
        .ENDC
;+
;   END OF GLOBALS.
;-
```

```
type dy:gdas3i.inc
!hco999
!hco99

;******************************************************************************;
;*  FILE: GDAS3I.INC                                                          *;
;*                                                                            *;
;*  NOTE: 1) THIS FILE CONTAINS ALL DATA DEFINITIONS FOR GDAS SAMPLING        *;
;*           PROGRAMS.  USE THE APPROPRIATE SAMPLE FLAG TO GENERATE           *;
;*           DATA FOR A SPECIFIED PROGRAM.                                    *;
;*        2) CURRENTLY VERSION 2.2 ONLY EXISTS FOR KENNEDY TAPE SAMPLING.     *;
;*                                                                            *;
;* CREATED  17-MAY-86  C.J.CENTER  Enhanced TPDEF.INC to create a general     *;
;*                                 data file for all sampling programs.       *;
;******************************************************************************;
        .PSECT  DATA
;+
;   SCAN HEADER AND ERC INFO - DON'T BREAK UP !!!
;-
SCANID: .WORD   072525  ; Start of scan flag.
ERCBUF: .BYTE   1       ; ERC time flag...
        .BYTE   2       ;   followed by ♦ words in ERC scan.
ERCMSB: .WORD   0       ; Storage for ERC time days & hours.
ERCLSB: .WORD   0       ;             ERC time minutes and seconds.
;+
;   SAMPLE DEFINITIONS....
;-
TIMPTR: .BLKW   2       ; ADDRESS OF START TIME IN FILE HEADER
GOTIME: .BLKW   2       ; ERC READOUT AT TRIGGER
CHMAP1: .BLKW   1       ; Bit map for the number of channels.
SAMBUF: .BLKW   16.     ; SAVE SPACE FOR SAMPLE BUFFER
;
NUMBER: .BLKW   7       ; 6 character buffer to hold user inputs.
NTEST:  .BLKW   1       ; Storage for number of scans.
NSCAN:  .BLKW   2       ; No. of A/D scans. (double precision).
SCANS:  .BLKW   2       ; Scan counter (single or double precision).
NCHAN:  .BLKW   1       ; No. of A/D channels to process for each scan.
;
CLKRAT: .BLKW   1       ; ERC clock rate (cycles/sec).
SAMRAT: .BLKW   1       ; No. of ERC cycles to skip when sampling.
BASRAT: .BLKW   2       ; Sample rate counters.
;+
; FORTRAN SUBROUTINE "TIMER" ARGUMENT BLOCK.
;-
DAY:    .BLKW   1       ; Address of days,
HOUR:   .BLKW   1       ;  hours,
MINUTE: .BLKW   1       ;   minutes,
SECOND: .BLKW   1       ;    and seconds.
ARGBLK: .WORD   4       ; 4 words in TIMER subroutine
        .WORD   DAY
        .WORD   HOUR
        .WORD   MINUTE
        .WORD   SECOND

;+
;    LSI 11/23 PRIORITIES....
;-
PR7:    .WORD   340     ; Priority 7.
PR5:    .WORD   240     ; Priority 5.
PR3:    .WORD   140     ; Priority 3.
PR0:    .WORD   0       ; Priority 0.
```

```
;    PROGRAM CONTROL....
;-
COMBUF: .BLKB   511.     ; Comment storage buffer.
CBUFEN: .BLKB   1        ; Last byte of comment storage.
LINBUF: .BLKB   80.      ; "GTLIN" line buffer storage.
AREA:   .BLKW   5        ; AREA FOR I/O EMT PARAMETERS
CAREA:  .BLKW   2        ; PARAMETERS FOR .GVAL & .SCCA PROGRAMMED REQ
SCCAST: .BLKW   1        ; STATUS WORD FOR .SCCA
TRIGON: .BYTE   0        ; Sample flag.
                         ;   > 0, sampling in progress.
;+                       ;   < 0, sampling complete.
;   D/A CALIBRATION FLAGS.
;-
PULSE:  .BLKB   1        ; (1=PULSE, 0=CONSTANT VOLTAGE)
DKCAL:  .BLKB   1        ; D/A Calibration flag.
                         ;      0, No calibration.
;+                       ;      1, Preform calibration.
;   DATA STORAGE
;-
        .EVEN
OUTPTR: .BLKW   1        ; POINTER TO OUTPUT BUFFER
OUTBUF: .BLKW   1        ; START ADDRESS OF OUTPUT BUFFER
OUTEND: .BLKW   1        ; END ADDRESS OF OUTPUT BUFFER
BLKNUM: .BLKW   1        ; STORAGE FOR OUTPUT FILE BLOCK NUMBER
;+
;   ERROR HANDLING MESSAGES.
;-
EREOF:  .BYTE   15,12
        .ASCIZ  " ERR......ATTEMPT TO WRITE OUTSIDE FILE LIMITS. "
EROPEN: .BYTE   15,12
        .ASCIZ  " ERR.........I/O CHANNEL NOT OPEN."
ERR05:  .BYTE   15,12
        .ASCIZ  / ERR......INVALID NUMBER OF CHANNELS.  /
ERRMEM: .BYTE   15,12
        .ASCIZ  / ERR......EXCEEDED AVAILABLE MEMORY SPACE. /
ERHARD: .BYTE   15,12
        .ASCIZ  / ERR......HARDWARE./
NOROOM: .BYTE   15,12
        .ASCII  / ERR......THE "PAD" IN FILE "GDAS3I.INC" MUST BE DECREASED. /
OFLOW:  .BYTE   15,12
        .ASCIZ  / WARNING.....DATA BUFFER OVERFLOW./
WAROUT: .BYTE   15,12
        .ASCIZ  / WARNING.....SCANS RECORDED > SCANS DESIRED.  /
WRERR:  .BYTE   15,12
        .ASCIZ  / ERR......FATAL WRITE. /
;+
;   QUERY USER INPUTS.
;-
GETCLK: .ASCIZ  " ERC FREQUENCY RATE (ERC Dial Setting) : "<200>
GETSAM: .ASCIZ  " ERC DIVISOR TO DETERMINE SAMPLE RATE  : "<200>
GETCHN: .ASCIZ  " NUMBER OF CHANNELS TO SAMPLE (decimal): "<200>
DAC.M1: .ASCIZ  " PREFORM D/A CALIBRATION ?       <Y,N> : "<200>
DAC.M2: .ASCIZ  " WILL A D/A PULSE BE INPUT ?     <Y,N> : "<200>
DAC.M3: .ASCIZ  " ENTER THE ADAC 1412 D/A COUNTS (octal): "<200>
```

```
;    DISPLAY USER INPUTS.
;-
INPUT:  .BYTE   15,12           ; Indicate the users inputs....
 .ASCIZ / ***************        DATA SAMPLING SPECIFICATIONS         ***************/
;
CHANM2: .BYTE   15,12           ; Message to indicate sample rate requested.
        .ASCII  / SAMPLING AT/
ARATE:  .BLKB   6               ; Ascii storage for sample rate...
PLUS:   .BYTE   40              ;   and "+" if divisor not even..
        .ASCIZ  " (DEC) SAMPLES/SEC."
CHANM1: .ASCIZ  / SAMPLING CHANNELS: /<200>
ACHSAM: .BLKB   60.             ; Storage for ascii channel map expansion...
TSCANS: .BLKB   12.             ; Storage for #scans in ascii.
;+
;    DATA FILE COMMENT MESSAGES.
;-
COMMNT: .BYTE 15,12
        .ASCII  / ENTERING COMMENTS    <Y,N> ?/<200>
CINFO:  .BYTE 15,12
   .ASCII  / ENTER UP TO 510 CHARACTERS OF TEXT./ <15><12>
   .ASCIZ  / A BLANK LINE INDICATES COMPLETION OF THE COMMENT SECTION./<15><12>
;
CPROMP: .BYTE   '>,200
COMEND: .ASCII  / YOU MAY ENTER/
CHLEFT: .BLKB   6
        .ASCIZ  / MORE CHARACTERS/
COMOVF: .ASCIZ  / BUFFER OVERFLOW - LAST LINE TRUNCATED/
;+
;    SAMPLING START MESSAGES.
;-
OKSAMP: .BYTE   15,12
        .ASCIZ  / TYPE <CR> TO BEGIN SAMPLING:/<200>
TRIGOU: .BYTE   15,12
        .ASCII /    ATTENTION:  A WARNING CHARACTER "W" IS PRINTED/<15><12>
        .ASCIZ /                WHEN INCOMPLETE SCANS ARE RECORDED/<15><12><12>
;+
;    SAMPLING FINISH MESSAGES.
;-
TTRIG:  .BYTE 15,12
        .ASCII /TRIGGERED AT /
AGOTIM: .BLKB 12.
        .BYTE 15,12
        .ASCII /END TIME/
        .BYTE 40,40,40,40,40
AENTIM: .BLKB 12.
        .BYTE 0
;**************************************************************************************************;
```

```
        .EVEN
;**********************************************************************;
;*      KENNEDY SAMPLING DEFINITIONS...                               *;
;**********************************************************************;
;+
;   DISPLAY PROGRAM HEADER INFORMATION.....
;-
HELLO:  .BYTE 15,12
 .ASCII " ***************        KENNEDY TAPE SAMPLING PROGRAM       ***************"
 .BYTE  15,12
 .ASCII / ***************      GDAS  VERSION 2.2  <JUNE 02,1986>      ***************/
 .ASCIZ / /<15><12>
;+
;   KENNEDY DATA MESSAGES....
;-
GETSCN: .ASCII  " NUMBER OF SCANS TO ACQUIRE      (octal): "<200>
SCANM1: .ASCII  / NUMBER OF SCANS TO SAMPLE (octal) : /<200>
SCANM2: .ASCII  / NUMBER OF SCANS RECORDED  (octal) : /<200>
QUERY:  .BYTE   15,12
        .ASCII  / REWIND KENNEDY TAPE  <Y,N> ?/<200>
;+
;   DATA STORAGE...
;-
        .EVEN
RECBUF: .BLKW   2000.       ; 2 Record buffer (1024. words) + overflow space.
RECNUM: .BLKW   1           ; Ouput file record number...
RECADD: .BLKW   1           ; Address of current data record.
RECEND: .BLKW   1           ; Addr. of end of record no. 1 & 2.
RECN0:  .BLKW   1           ; Base address of record no. 0.
RECN1:  .BLKW   1           ; Base address of record no. 1.
RECFLG: .BYTE   0           ; .EQ. 1: Waiting for complete record of data.
                            ;     -1: Complete record has been data recorded.

        .EVEN
;**********************************************************************;
        .ENDC               ; END:       KENNEDY TAPE CONDITIONALS.
        .IF  NE  FSAMP      ; START:     FOREGROUND/BACKGROUND CONDITIONALS.
        .EVEN
;**********************************************************************;
;*      FOREGROUND / BACKGROUND SAMPLING SPECIFICS.                   *;
;**********************************************************************;
;+
;   DISPLAY PROGRAM HEADER INFORMATION.....
;-
HELLO:  .BYTE 15,12
 .ASCII " ***************  FOREGROUND / BACKGROUND SAMPLING PROGRAM ***************"
        .BYTE  15,12
 .ASCII / ***************      GDAS  VERSION 2.2  <JUNE 02,1986>      ***************/
        .ASCIZ / /<15><12>
;+
;   DATA FILE NAME (INPUT FROM USER)....
;-
ENTER:.BYTE 15,12

      .ASCII /...PLEASE ENTER OUTPUT FILE NAME (WITHOUT EXTENSION).../<15><12>
      .ASCIZ /     *FILE*          <CR>    ...WRITE FILE TO DISK./<15><12>
SCANM1: .ASCII  / NUMBER OF SCANS TO SAMPLE (decimal) : /<200>
SCANM2: .ASCII  / NUMBER OF SCANS RECORDED  (decimal) : /<200>
GETSCN: .ASCII  " NUMBER OF SCANS TO ACQUIRE   (decimal): "<200>
```

```
;   FILE STORAGE INFORMATION...
;-
        .EVEN
SPTEMP: .BLKW   1        ; Temp. stack ptr. storage during .CSIGEN command.
DEFEXT: .RAD50 "DAT"     ;DEFAULT EXTENSION FOR INPUT FILE
        .RAD50 "RAW"     ;DEFAULT EXTENSION FOR OUTPUT FILE
        .WORD 0
        .WORD 0
DEVHND: .WORD 0          ;DEVICE HANDLERS GO HERE WHEN NEEDED
;+
; The next area is opened for A/D data file storage.  As defined by .CSIGEN,
; a file is opened in the first available memory location following DEVHND.
;-
PAD:    .=.+100000       ; Leave room for device handlers.
ENDPAD: .
        .ENDC            ; End of Foreground / Background conditionals.
        .IF  NE  RSAMP
        .EVEN
;*************************************************************************;
;*      BUBBLE, FLOPPY, DISK SAMPLING.                                   *;
;*************************************************************************;
;+
;   DATA OUTPUT BUFFER POINTER DEFINITIONS...
;-
BLKADD: .BLKW   1        ; Address of current data block.
BLKN0:  .BLKW   1        ; Base address of block no. 0.
BLKN1:  .BLKW   1        ; Base address of block no. 1.
BLKEND: .BLKW   1        ; Address ptr. for end of comment block.
BLKFLG: .BLKB   0        ; .EQ. 1: Waiting block of data to fill up.
                         ;     -1: Block of data recorded.
;+
;   DISPLAY PROGRAM HEADER INFORMATION.....
;-
HELLO:  .BYTE 15,12
 .ASCII " **************          I/O SAMPLING PROGRAM          **************"
 .BYTE  15,12
 .ASCII / **************      GDAS  VERSION 2.2  <JUNE 02,1986>      **************/
 .ASCIZ / /<15><12>
;
SCANM1: .ASCII  / NUMBER OF SCANS TO SAMPLE (decimal) : /<200>
SCANM2: .ASCII  / NUMBER OF SCANS RECORDED  (decimal) : /<200>
GETSCN: .ASCII  " NUMBER OF SCANS TO ACQUIRE   (decimal): "<200>
;+
;   RT SAMPLE ERROR HANDLING...
;-
WAITER: .BYTE   15,12
        .ASCIZ  " ERR......ASYNCHRONOUS I/O ERROR."
;+
;   DATA FILE NAME(S) (INPUT FROM USER)....
;-
ENTER:.BYTE 15,12
       .ASCII /...PLEASE ENTER OUTPUT FILE NAME (WITHOUT EXTENSION).../<15><12>
       .ASCII /    *FILE=            <CR>    ...WRITE FILE TO DISK./<15><12>
       .ASCII /    *DY:FILE[974]=  <CR>    ...WRITE FILE TO FLOPPY./
       .ASCII / (974 BLKS MAXIMUM)./<15><12>
       .ASCII /    *BY:FILE[1962]= <CR>    ...WRITE FILE TO BUBBLE MEMORY/
       .ASCIZ / (1962 BLKS MAXIMUM)./<15><12><12>
```

```
;     I/O FILE HANDLING RESERVATIONS...
;-
        .EVEN
SPTEMP: .BLKW   1       ; Temp. stack ptr. storage during .CSIGEN command.
QAREA:  .BLKW   10      ; Extra Q-element entry area.
DEFEXT: .RAD50 "DAT"    ;DEFAULT EXTENSION FOR INPUT FILE
        .RAD50 "RAW"    ;DEFAULT EXTENSION FOR OUTPUT FILE
        .WORD 0
        .WORD 0
DEVHND: .               ;DEVICE HANDLERS GO HERE WHEN NEEDED
;+
; The next area is opened for A/D data file storage.  As defined by .CSIGEN,
; a file is opened in the first available memory location following DEVHND.
;-
PAD:    .=.+10000
ENDPAD: .
;************************************************************************;
        .ENDC                   ; END:    BUBBLE MEMORY CONDITIONALS.
        .IF  NE  XSAMP          ; START:  EXTENDED MEMORY CONDITIONALS.
        .EVEN
;************************************************************************;
;*   EXTENDED MEMORY SAMPLE PROGRAM.                                     *;
;************************************************************************;
;+
;    DISPLAY PROGRAM HEADER INFORMATION.....
;-
HELLO:  .BYTE 15,12
 .ASCII " ****************  EXTENDED MEMORY SAMPLING PROGRAM  ****************"
 .BYTE  15,12
 .ASCII / *************     GDAS  VERSION 2.2  (JUNE 02,1986)     *************/
 .ASCIZ / /<15X12>
;+
;    DATA FILE NAME (INPUT FROM USER)....
;-
ENTER:.BYTE 15,12
      .ASCII /...PLEASE ENTER OUTPUT FILE NAME (WITHOUT EXTENSION).../<15X12>
      .ASCIZ /     *FILE*          <CR>    ...WRITE FILE TO DISK./<15X12>
SCANM1: .ASCII  / NUMBER OF SCANS TO SAMPLE (decimal) : /<200>
SCANM2: .ASCII  / NUMBER OF SCANS RECORDED  (decimal) : /<200>
GETSCN: .ASCII  " NUMBER OF SCANS TO ACQUIRE   (decimal): "<200>
;+
;    EXTENDED MEMORY ERROR HANDLING...
;-
ERR:    .BYTE   15,12
        .ASCIZ  " ERR.........XM - ERROR "
ERRNO:  .ASCIZ  /00/
;+
;    DATA STORAGE
;-
        .EVEN
WNDCNT: .WORD   23.     ; Maximum window remappings.
INOFF:  .BLKW   1       ; Initial offset into region.
SAMPFL: .BYTE   0       ; Sample flag:    0, wait.  Else sample.
ERCFLG: .BYTE   0       ; ERC clock flag: 0, wait.  Else read time.
;
XAREA:  .BLKW   2       ; AREA FOR EXTENDED MEMORY INFO..
XMBUF:  .RDBBK  3072.   ; BIGGEST PHYSICAL REGION AVAILABLE (96K).
WINDOW: .WDBBK  2,128.,,0,0,WS.MAP      ; WINDOW (INITIAL OFFSET=0)
```

```
;   FILE STORAGE INFORMATION...
;-
SPTEMP: .BLKW   1       ; Temp. stack ptr. storage during .CSIGEN command.
DEFEXT: .RAD50 "DAT"    ;DEFAULT EXTENSION FOR INPUT FILE
        .RAD50 "RAW"    ;DEFAULT EXTENSION FOR OUTPUT FILE
        .WORD 0
        .WORD 0
DEVHND: .WORD 0         ;DEVICE HANDLERS GO HERE WHEN NEEDED
;+
; The next area is opened for A/D data file storage.  As defined by .CSIGEN,
; a file is opened in the first available memory location following DEVHND.
;-
PAD:    .=.+10000
ENDPAD: .
        .ENDC           ; End of Extended memory conditionals.
;*************************** END OF DATA DEFINITIONS ***************************
```

```
type dy:gdas4i.inc
!hcol

;****************************************************************************;
;*   INCLUDE FILE:  GDAS4I.INC                                              *;
;*                                                                          *;
;*   CREATED   05-JUN-86   C.J.CENTER    GDAS MACRO include file.           *;
;****************************************************************************;
;*   MACRO:      SAVREG                                                     *;
;*   PUSHES REGISTERS ONTO THE STACK.                                       *;
;****************************************************************************;
        .MACRO  SAVREG
        MOV     R0,-(SP)
        MOV     R1,-(SP)
        MOV     R2,-(SP)
        MOV     R3,-(SP)
        MOV     R4,-(SP)
        MOV     R5,-(SP)
        .ENDM
;****************************************************************************;
;*   MACRO:      RESREG                                                     *;
;*   POPS REGISTERS OFF THE STACK.                                          *;
;****************************************************************************;
        .MACRO  RESREG
        MOV     (SP)+,R5
        MOV     (SP)+,R4
        MOV     (SP)+,R3
        MOV     (SP)+,R2
        MOV     (SP)+,R1
        MOV     (SP)+,R0
        .ENDM
```

```
type dy:prinit.mac
!hco9

        .TITLE  PRESTON INITIALIZATION.
        .GLOBL  PRINIT
;***************************************************************************;
;*                              WESTON OBSERVATORY                          *;
;*                        DEPT. OF GEOLOGY AND GEOPHYSICS                   *;
;*                     DATA ACQUISITION SYSTEM:  VERSION 2.0                *;
;*                                                                          *;
;*  FILE: PRINIT.MAC                                                        *;
;*                                                                          *;
;*  Creation  10-nov-85   c.j.center   Updated A/D programming steps.      *;
;*  Revision  21-nov-85   c.j.center   Installed as a GLOBAL.              *;
;*            29-dec-85   c.j.center   Updated DKINIT.MAC for XM routines. *;
;*            16-jan-85   c.j.center   Install TABLE to allow relative DMA *;
;*                                      addressing.                         *;
;*            17-feb-86   c.j.center   Installed A/D error handling.       *;
;*            03-mar-86                Change clock divisor.               *;
;* THIS SUBROUTINE.......                                                   *;
;*   -1-  RESETS THE A/D.                                                   *;
;*   -2-  SETS UP THE REQUIRED "DATI" FUNCTION BITS.                        *;
;*   -3-  ENTERS A CONTROL WORD TO ENTER THE FOLLOWING PROGRAM MODES...     *;
;*          CLOCK DIVISIOR: #CLKDIV                                         *;
;*          FIRST CHANNEL:  #0                                              *;
;*          LAST CHANNEL:   NCHAN (global)                                  *;
;*          RSO-STOP CONDITION.                                             *;
;*   -5-  EXITS ON ERROR.                                                   *;
;***************************************************************************;
;+
;  DEFINE GLOBALS AND DRV11B BUS ADDRESSES....
;-
        .MCALL  .PRINT,.EXIT
        .GLOBL  NCHAN
BASE = 172410                   ; DRV11B base address.
DRVWCR = BASE                   ; Word Count Register
DRVBAR = BASE+2                 ; Bus Address Register.
DRVCSR = BASE+4                 ; Control Status Register.
CLKDIV = 74.                    ; Minimal PRESTON A/D clock divisor.
; Misc...
        .PSECT  DATA
TABLE:  .BLKW   20.             ; DMA storage table...
ERR:    .ASCIZ  "  ERR................PRESTON A/D NOT FUNCTIONING."

        .PSECT  PROG
PRINIT: MOV     R0,-(SP)        ; SAVREG....
        MOV     R1,-(SP)
        MOV     R2,-(SP)
        MOV     R3,-(SP)
        MOV     R4,-(SP)

        MOV     #DRVWCR,R1      ; R1 --> DRV11B Word Count Register.
        MOV     #DRVBAR,R2      ; R2 --> DRV11B Buffer Address Register.
        MOV     #DRVCSR,R3      ; R3 --> DRV11B Control/Status Register.
        MOV     #TABLE,R4       ; R4 --> Memory location for DRV11B DMA DATI.
        MOV     #177777,(R4)+   ; Store RESET to clear out A/D.
        MOV     #20470,(R4)+    ; Store "CONTROL WORD" to send the following..
        MOV     #CLKDIV,(R4)+   ;   1. "CLOCK DIVISOR".
        MOV     #0,(R4)+        ;   2. "FIRST CHANNEL"
        MOV     #NCHAN,(R4)+    ;   3. "LAST CHANNEL".
        MOV     #200,(R4)       ;   4. "RSO-STOP" condition.

        CLR     @R1             ; Assure READY is HI.
        MOV     #-6,@R1         ; No. of words to strobe in.
        MOV     #TABLE,@R2      ; BAR --> TOP of TABLE.
        BIS     #6,@R3          ; Set FUNCT1=1, FUNCT2=1
        BIC     #4,@R3          ; Set FUNCT1=1, FUNCT2=0

        MOV     #35.,R0         ; 100+ microsecond delay for A/D after last
        SOB     R0,.            ;   function strobed in.
        MOV     #403,@R3        ; Set CYCREQ and GO bit.
;+
;   ERR ??? TEST START UP PROCEDURE....
;-
        MOV     #1000.,R0       ; Assign a counter as a timer...
10$:    DEC     R0              ;   and decrement.
        BEQ     ERROR           ; If words not strobed in yet...then ERR.
        TST     @R1             ; Test DRVWCR for all words being
        BNE     10$             ;   strobed in before returning...

        MOV     (SP)+,R4        ; RESREG....
        MOV     (SP)+,R3
        MOV     (SP)+,R2
        MOV     (SP)+,R1
        MOV     (SP)+,R0
        RTS     PC              ;   AND RETURN.
;+
;  FATAL ERROR ON PRESTON A/D...
;-
ERROR:  .PRINT  #ERR            ; Print the error..
        ADD     #10.,SP         ; Restore the stack pointer,
        .EXIT                   ;   and exit.

        .END    PRINIT
```

```
type dy:gdas3m.mac

!hhco
!hco10

.       .TITLE  GDAS3M
        .IDENT  /V02.20/
;****************************************************************************;
;*                           WESTON OBSERVATORY                             *;
;*                    DEPT. OF GEOLOGY AND GEOPHYSICS                       *;
;*       GEOPHYSICAL DATA ACQUISITON SYSTEM (GDAS) SOFTWARE LIBRARY         *;
;*  FILE:  GDAS3M.MAC                                                       *;
;*                                                                          *;
;*  CREATED  05-JUN-86  C.J.CENTER  This file contains all subroutines for  *;
;*                                    the GDAS system.                      *;
;*  REVISED  06-AUG-86  C.J.CENTER  Add titles and globals for GDAS library.*;
;*                                                                          *;
;*  NOTE:  FILE "GDAS1I.INC" CONTAINS THE "IF" CONDITIONALS TO COMPILE      *;
;*         PROGRAMS FOR RT11-FB & RT11-XM OPERATING SYSTEMS.                *;
;*                                                                          *;
;*  SUBROUTINE LIST:                                                        *;
;*      DAC:    Query user for D/A calibration & set DKCAL flag.            *;
;*      INTON:  Initialize system interrupts.                               *;
;*      INTOFF: Stop system interrupts.                                     *;
;*      INTKIL: Stop undesireable system interrupts.                        *;
;*      TYSAMP: Display's sample rate on terminal.                          *;
;*      ERCINT: ERC interrupt B handler.                                    *;
;****************************************************************************;
        .MCALL  .SETTOP,.GVAL,.CRRG,.CRAW,.MAP,.UNMAP
        .MCALL  .CSIGEN,.WRITW,.WAIT,.CLOSE,.PRINT,.GTLIN,.TTYIN,.EXIT

        .INCLUDE    \GDAS1I.INC\        ; Equivalences & conditionals.
        .INCLUDE    \GDAS2I.INC\        ; Globals.
        .INCLUDE    \GDAS4I.INC\        ; Macros.
        .PSECT  PROG
        .TITLE  ELAPSE
        .GLOBL  ELAPSE
;******************************************************************;
;*  PROGRAM CALL:   JSR   PC,ELAPSE                              *;
;*    THIS SUBROUTINE RETURNS AFTER AN ELAPSED TIME HAS EXPIRED. *;
;*                                                               *;
;*  CREATED  13-AUG-86  C.J.CENTER                               *;
;******************************************************************;
        .GLOBL  SECOND,MINUTE,HOUR,ARGBLK,TIMER,ERCCSR
ELAPSE: SAVREG                          ; Save registers.
M.NEXT = 5      ; Minute elapse time.
H.NEXT = 0      ; Hour elapse time
;+
; SET POINTERS.
;-
        MOV     SECOND,R1       ; R1 --> Seconds of last read.
        MOV     MINUTE,R2       ; R2 --> Minutes of last read.
        MOV     HOUR,R3         ; R3 --> Hour of last read.
        MOV     #ARGBLK,R5      ; R5 --> Fortran argument block pointer.
;+
; GET RESTART TIME.
;-
        ADD     #M.NEXT,R2      ; Set minute time to resample.
        CMP     #60.,R2         ; Minutes exceed 60 ?
        BHI     HRS             ;  NO - continue.
        SUB     #60.,R2         ;  YES- carry minutes into
        INC     R3              ;         hours.
HRS:    ADD     #H.NEXT,R3      ; Set hour time to resample.
        CMP     #24.,R3         ; Hours exceed 24 ?
        BHI     1000$           ;  NO - continue.
        SUB     #24.,R3         ;  YES- carry hours into
;       INC     R4              ; days.  "but not really needed."
;
1000$:  JSR     PC,TIMER        ; Read current time.
        CMP     @10(R5),R1      ; Seconds match ?
        BNE     1000$           ;  NO.
        CMP     @6(R5),R2       ; Minutes match ?
        BNE     1000$           ;  NO.
        CMP     @4(R5),R3       ; Hours match ?
        BNE     1000$           ;  NO.
;
        MOV     #40,@#ERCCSR    ; Restore ERC interrupts.
        RESREG                  ; Restore registers
        RTS     PC              ;   and return.
;
```

```
;**************************************************************************;
;*  SUBROUTINE CALL:    JSR  PC,DASTRT                                   *;
;*                                                                       *;
;*  CREATED  10-JUN-86  C.J.CENTER   Query user for D/A calibration.     *;
;*  REVISED  06-AUG-86  C.J.CENTER   Acquire voltage input from user.    *;
;*           25-SEP-86               Remove pulse,  init "DAVOLT".       *;
;**************************************************************************;
        .GLOBL  DAVOLT,DASTRT,IVOLTS,DAC.M3,DKFLG
;
        .PSECT  DATA
IVOLTS: .BLKW   1        ; ADAC D/A voltage count.
DAVOLT: .BLKW   1        ; Voltage in the D/A.
DKFLG:  .BLKW   1        ; When decreased to 0,  voltage is sent to the A/D.
;+
; ENTER.
;-
        .PSECT  PROG
DASTRT: MOV     R0,-(SP)        ; SAVREG...
        MOV     R5,-(SP)
        CLRB    DKCAL           ; Initialize DKCAL to zero.
;+
; QUERY USER FOR CALIBRATION.
;-
        MOV     #LINBUF,R0      ; R0 --> input string buffer.
135$:   .GTLIN  R0,#DAC.M1      ; Calibration ??
        CMPB    @R0,#'N         ;   NO -
        BEQ     160$            ;     exit.
        CMPB    @R0,#'Y         ;  YES - continue
        BNE     135$            ; Else ask again.
;+
; SET CALIBRATION FLAGS.
;-
        CLR     DAVOLT          ; Init the D/A volt to zero.
        INCB    DKCAL           ; Turn on the calibration flag (DKCAL).
        CLRB    PULSE           ; Init. the pulse flag to zero.
        CLR     @#DAC1          ; Init. D/A voltage (DAC1) to zero.
        MOV     #256.,DKFLG     ; D/A input voltage starts at 256. scans.
;+
; PULSE OR CONSTANT VOLTAGE ?
;-
;140$:  .GTLIN  R0,#DAC.M2      ; Will user input a pulse ?
;       CMPB    @R0,#'N         ;   NO...
;       BEQ     150$            ;     or
;       CMPB    @R0,#'Y         ;       YES ?
;       BNE     140$            ; Assure answer is (Y or N).
;       INCB    PULSE           ; Set PULSE flag on.
;+
; QUERY USER FOR INPUT VOLTAGE.
;-
150$:   .PRINT  #DAC.M3         ; Query user for ADAC D/A counts.
        JSR     PC,GETOCT       ; Translate octal ascii to octal numbers
        MOV     R3,IVOLTS       ;   and store here.
;+
; EXIT.
;-
160$:   MOV     (SP)+,R0        ; RESREG..
        MOV     (SP)+,R5        ;   and
        RTS     PC              ;     return.
;
```

```
;**********************************************************************
;* PROGRAM CALL:  JSR  PC,DASTOP                                      *
;*                                                                    *
;* CREATED  06-AUG-86  C.J.CENTER  Clear the ADAC 1412 D/A registers. *
;**********************************************************************
        .GLOBL  DASTOP
DASTOP: CLR     @#DAC1           ; Clear voltage in D/A
        RTS     PC               ;   and return.
;
        .TITLE  INTKIL
        .GLOBL  INTKIL
;************************************************************************;
;*  PROGRAM CALL:  JSR  PC,INTKIL                                       *;
;*    MAINTAINS UNDESIREABLE INTERRUPTS DORMIT.                         *;
;*                                                                      *;
;*  CREATED  11-AUG-86  C.J.CENTER                                      *;
;************************************************************************;
INTKIL: MOV     #2,ERCVEC+2       ; RETURN IF
        MOV     #ERCVEC+2,ERCVEC ;  ERC REQ A INTERRUPTS.
        CLR     ERCCSR            ; ERC INTERRUPT ENABLE OFF
        RTS     PC                ; Return.
;
        .TITLE  CLKTIM
        .GLOBL  CLKTIM
;****************************************************************;
;*  PROGRAM CALL:  JSR  PC,CLKTIM                               *;
;*    START THE ERC CLOCK.                                      *;
;*                                                              *;
;*  CREATED  11-AUG-86  C.J.CENTER                              *;
;****************************************************************;
CLKTIM: MOV     R0,-(SP)          ; SAVREG.
        MOV     #ERCCSR,R0        ; Get ERC clock address.
        BISB    #2,@R0            ; Latch and read ERC
        MOV     ERCIN,ERCLSB      ;   minutes and seconds.
        INCB    @R0               ; Latch and read ERC
        MOV     ERCIN,ERCMSB      ;   days and hours.
        MOV     #40,@R0           ; Restore ERC INT B after time read.
        MOV     (SP)+,R0          ; RESREG and
        RTS     PC                ;  exit.
;
        .TITLE  INTON
        .GLOBL  INTON
;****************************************************************;
;*  PROGRAM CALL:  JSR  PC,INTON                                *;
;*    INITIALIZE THE SAMPLING INTERRUPTS.                       *;
;*                                                              *;
;*  CREATED  11-AUG-86  C.J.CENTER                              *;
;****************************************************************;

;
INTON:  MOV     R0,-(SP)         ; SAVREG...
        MTPS    PR7              ; DON'T INTERUPT - TURN ON SAMPLE CLOCK.
        MOV     #ERCINT,ERCVEC+4 ; LOAD ERC REQ B VECTOR
        MOV     PR7,ERCVEC+6      ; DON'T INTERRUPT ERC SERVICE ROUTINE
        MOV     #40,ERCCSR          ; Start ERC clock interrupts
;
        MOV     #CLKVEC,R0         ; ADDRESS OF SAMPLE CLOCK VECTORS
        MOV     #SAMPLE,(R0)+      ; GO HERE WHEN CLOCK INTERUPTS
        MOV     PR7,@R0            ; CLOCK WILL HAVE HIGHEST PRIORITY
        MTPS    PR0                ; Allow interrupt servicing.
        MOV     (SP)+,R0           ; RESREG.
        RTS     PC                 ; Return.
;
        .GLOBL  INTOFF
;**********************************************************************;
;*  PROGRAM CALL:   JSR  PC,INTOFF                                   *;
;*    STOP ERC CLOCK INTERRUPTS.                                     *;
;*                                                                   *;
;*  CREATED  11-AUG-86                                               *;
;**********************************************************************;
INTOFF: MTPS    PR7                  ; Raise processor priority.
        MOV     #RTI,CLKVEC+2        ; IGNORE SAMPLE CLOCK
        MOV     #CLKVEC+2,CLKVEC
        CLR     ERCCSR                 ; Turn off ERC clock interrupts
        MOV     #2,ERCVEC+6            ;   and disable
        MOV     #ERCVEC+6,ERCVEC+4     ;     ERC interrupt.
        RTS     PC                     ; Return.
;
        .TITLE  TYSAMP
;****************************************************************************;
;*   SUBROUTINE:  TYSAMP                                                    *;
;*   TYPE THE SAMPLE RATE THE USER REQUESTED.                               *;
;****************************************************************************;
        .GLOBL  TYSAMP
TYSAMP: MOV     R0,-(SP)            ; SAVREG...
        MOV     R1,-(SP)
        MOV     R2,-(SP)
        MOV CLKRAT,R1               ;GET SAMPLE CLOCK FREQ
        CLR R0                      ;MAKE IT DOUBLE PRECISION
        MOV SAMRAT,R2               ;GET "SAMPLE RATE" COUNTER
        MOV R2,BASRAT               ;STORE IT THERE
        MOV R2,BASRAT+2             ;  AND THERE
        DIV R2,R0                   ;DETERMINE REAL SAMPLE RATE
        TST R1                      ;REMAINDER?
        BEQ 14$                     ;BRANCH IF NO
        MOVB #'+,PLUS               ;INDICATE THERE'S A LITTLE MORE
14$:    MOV R0,R1                   ;SAMPLE RATE
        JSR R0,OTOAD                ;  TO ASCII
        .WORD ARATE                 ;    THERE
        .PRINT  #CHANM2             ; Message for the sample rate.
        MOV     (SP)+,R2            ; RESREG
        MOV     (SP)+,R1            ;   and..
        MOV     (SP)+,R0
        RTS     PC                  ;     return..
;
```

```
;
        .TITLE  TYSCAN
;*************************************************************************;
;*   SUBROUTINE:  TYSCAN                                                *;
;*   TYPE THE NUMBER OF SCANS INPUT BY THE USER.                        *;
;*************************************************************************;
        .GLOBL  TYSCAN
TYSCAN: MOV NSCAN,R1                ;# OF DATA SCANS TO ACQUIRE
        JSR R0,OTOAD                ;CONVERT TO ASCII
        .WORD TSCANS                ;THERE
        .PRINT  #SCANM1             ; Print the number of
        .PRINT  #TSCANS             ;   scans.
        RTS     PC
;

!hhco
!hco10

.        .TITLE  TYCMAP
        .GLOBL  TYCMAP
;*****************************************************************;
;*  CALL:    JSR   PC,TYCMAP                                    *;
;*  FUNC:    1. CREATE A CHANNEL BIT MAP.                       *;
;*           2. TYPE THE CHANNEL NUMBERS BEING SAMPLED.         *;
;*****************************************************************;
        .GLOBL  NCHAN
;
;
TYCMAP: MOV CHMAP1,R0               ;GET BIT MAP FOR ANALOG CHANNELS TO SAMPLE
;
        MOV #ACHSAM,R2              ;ADDR OF ASCII STRING FOR CHANNEL NUMBERS
        CLR R1                      ;CHANNEL 0 FIRST
10$:    ROR R0                      ;SAMPLING THIS CHANNEL
        BCC 12$                     ;BRANCH IF NO
        INC     R1                  ; Allow channel # to be relative to 1.
        JSR R0,OTOAD                ;CONVERT CHAN NUMBER TO ASCII
        .WORD ARATE                 ;TEMP STORAGE THERE
        DEC     R1                  ; Restore R1 to 0 relative.
        MOVB #' ,(R2)+              ;MOVE A SPACE TO STRING
        MOVB ARATE+4,(R2)+          ;ASCII CHANNEL NUMBER TO STRING
        MOVB ARATE+5,(R2)+
12$:    INC R1                      ;NEXT CHANNEL
        CMP R1,NCHAN2               ;DONE FOR ALL CHANNELS IN A/D?
        BLT 10$                     ;BRANCH IF NO
;
        CLRB @R2                    ;TERMINATE STRING WITH A ZERO BYTE
        .PRINT  #CHANM1             ; Print the channel map
        .PRINT  #ACHSAM             ;   requested by the user.
        RTS     PC
;
```

```
;******************************************************************************;
;*  SUBROUTINE:  GETCOM                                                       *;
;*  GET A BLOCK OF COMMENTS FROM THE USER.                                    *;
;******************************************************************************;
        .GLOBL  GETCOM
GETCOM: SAVREG                          ; Save registers.
;+
;   1st...CLEAR THE BLOCK...
;-
        MOV     #COMBUF,R1              ; R1 --> base of comment storage buffer.
        MOV     R1,R5                   ; R5 --> base of comment storage buffer.
        CLRB    (R5)+                   ; Initialize 1st byte in buffer to zero.
        MOV     #256.,R0                ; 256. words in a block.
ZAP:    CLR     (R1)+                   ; And clear the block buffers
        SOB     R0,ZAP                  ;   contents.
;+
;   2nd...DOES USER WANT COMMENTS ???
;-
        MOV #LINBUF,R0                  ;POINT TO START OF INPUT STRING
20$:    .GTLIN R0,#COMMNT               ;ASK USER IF HE WANTS TO ENTER COMMENTS
        CMPB @R0,#'N                    ;DID HE RESPOND NO?
        BEQ 30$                         ;BRANCH IF NO
        CMPB @R0,#'Y                    ;DID HE RESPOND YES?
        BNE 20$                         ;TRY AGAIN IF HE DIDN'T
;+
;   3rd...OBTAIN COMMENTS IF USER DESIRES...
;-
        .PRINT  #CINFO                  ; Instruct user on entering comments.
22$:    MOV     #LINBUF,R0              ; R0 --> line buffer.
        .GTLIN  R0,#CPROMP              ; Prompt each comment line with a ">".
24$:    MOVB (R0)+,(R5)+                ; Move comments to buffer byte by byte.
        BEQ 26$                         ;ZERO BYTE MEANS END OF STRING
        CMP R5,#CBUFEN                  ;AT END OF COMMENT BUFFER?
        BLO 24$                         ;BRANCH IF NOT AT END
        BR 28$                          ;BUFFER OVERFLOW - BRANCH
26$:    DEC R0                          ;POINT TO THE ZERO BYTE
        CMP R0,#LINBUF                  ;FIRST CHARACTER OF LINE?
        BEQ 30$                         ;DONE IF IT IS FIRST
        MOVB #15,-1(R5)                 ;MOVE A CARRIAGE RETURN OVER THE ZERO BYTE
        MOVB #12,(R5)+                  ;THEN A LINE FEED
        MOV #CBUFEN,R1                  ;ADDRESS BUFFER'S END
        SUB R5,R1                       ;DETERMINE # BYTES IN BUFFER
        CMP R1,#120.                    ;MORE THAN 120 BYTES LEFT IN BUFFER?
        BGT 22$                         ;GET ANOTHER LINE IF SO
        JSR R0,OTOAD                    ;CONVERT # BYTES REMAINING TO ASCII
        .WORD CHLEFT                    ;PUT ASCII THERE
        .PRINT #COMEND                  ;TELL USER HOW MANY BYTES HE HAS REMAINING
        BR 22$                          ;GO GET NEXT LINE
28$:    .PRINT #COMOVF                  ;PRINT MESSAGE FOR COMMENT BUFFER OVERFLOW
        CLRB CBUFEN                     ;END OF COMMENTS AT END OF BUFFER
;
30$:    RESREG                          ; Restore registers
        RTS     PC                      ;   and return.
```

```
;****************************************************************************;
;*  CALL:     JSR    PC,WRTHED                                              *;
;*  Write and assign pointers for the header information in the first block *;
;*  of the output buffer.                                                   *;
;*  Inputs:                                                                 *;
;*      OUTBUF:  Ptr. to base address of output data buffer.                *;
;*      OUTPTR:  Ptr. to current open location in output buffer. (updated   *;
;*               in this subroutine).                                       *;
;****************************************************************************;
        .GLOBL  WRTHED
WRTHED: MOV     R0,-(SP)        ; SAVREG...
        MOV     R1,-(SP)
        MOV     OUTBUF,R0       ; Get the base address of the output buffer...
        MOV     R0,R1           ; Update the output buffer
        ADD     #512.,R1        ;   pointer to the start
        MOV     R1,OUTPTR       ;     of the next block.
;+
;   BLOCK #1:  SAMPLE HEADER DEFINITIONS.
;-
        CLR     (R0)+           ; WORD 1:  ZERO.
        MOV     CLKRA 3:  ERC CLOCK FREQUENCY
;*
; GET ERC TIME.2,-4,-4,' ,-2,-4,':,0,-4,-4,':,-4,
;-
        .PSECT  PROG
ASCRET: RESREG                  ; Restore registersister.A WORD
BITMAP: SAVREG
15$:    MOV     R5,CH
        INC R2                  ;WE DON'T HAVE TO
4$:     SOB R4,2$
        ADD #60,R1              ;REMAINDER IS LAST DIGIT
        MOVB R1,(R5)+
REGRET: RESREG                  ; Restore registers...
        TST     (R0)+           ; Adjust R0 for proper return address
        RTS     R0              ;   and RETURN VIA R0.
TENS:   .WORD 10.,100.,1000.,10000.,177777
;
```

```
;*************************************************************************;
;*   SUBROUTINE GETOCT                                                   *;
;*   NOTE:  This subroutine receives DP octal numbers from the terminal and  *;
;*          stores their ascii equivalent in registers R0 and R1.          *;
;*   REGISTER DEF...                                                     *;
;*                  R3:  Low order DP word  or SP word.                  *;
;*                  R2:  High order DP word or clear.                    *;
;*************************************************************************;
        .GLOBL  GETOCT
GETOCT: MOV     R0,-(SP)        ; SAVREG....
        MOV     R1,-(SP)
        MOV     R4,-(SP)
        CLR     R1              ; Clear temporary storage register...
        CLR     R2              ; Clear DP HI order word.
        CLR     R3              ; Clear DP LO order word.
;
100$:   .TTYIN                  ; Get the number.
        CMPB    R0,#15          ; If no characters entered clear
        BEQ     100$            ;   the CR,
        CMPB    R0,#12          ;     and the LF,
        BEQ     DONE            ;       and exit.
;
        SUB     #60,R0          ; Convert ASCII code to decimal number.
        BLO     120$            ; Assure character entered is
        CMPB    R0,#10          ;   in the octal range.
        BHIS    120$            ; Too high?  Ask for another character.
;+
;  DP SHIFT....
;       Store octal equivalent in DP...
;       R2 = HI order byte.,  R3 = LO order byte.
;-
        MOV     #3,R4           ; 3 shifts..
110$:   ASL     R2
        ASL     R3
        ADC     R2
        SOB     R4,110$
;
        ADD     R0,R3           ; Add current TTY buffer value to LO order
        ADC     R2              ;   and include carry...
        BR      100$            ; Get next character.
;+
;   INVALID OCTAL CHARACTER HANDLING....
;-
120$:   .TTYIN                  ; ERR handling...
        CMPB    R0,#12          ;   Clear the input data
        BNE     120$            ;     on the TTYIN line.
        .PRINT  #WARN           ; Indicate ERR to user,
        CLR     R2              ; Reset DP HI order word
        CLR     R3              ;   and DP LO order word
        BR      100$            ;     and ask for more data....
;
DONE:   MOV     (SP)+,R4        ; RESREG...
        MOV     (SP)+,R1
        MOV     (SP)+,R0        ;   and..
        RTS     PC              ;     return to calling program.
;
```

```
;***************************************************************************;
;*  SUBROUTINE PUTOCT.                                                     *;
;*  Created  28-apr-86  c.j.center  See note.                              *;
;*                                                                         *;
;*  NOTE:  This subroutine converts single or double precision data into   *;
;*          its ascii equivalent for terminal printout.                    *;
;*  INPUT REGISTERS  R3,R2 "REQUIRED".                                     *;
;*      R3 --> LO order DP or SP.                                          *;
;*      R2 --> HI order DP or "CLEARED".                                   *;
;*      R5 --> Output ascii address.                                       *;
;***************************************************************************;
        .GLOBL  PUTOCT
PUTOCT: MOV     R0,-(SP)            ; SAVREG...
        MOV     R1,-(SP)
        MOV     R4,-(SP)
        MOVB    #2,DPFLAG           ; Initialize DP flag for DP operations....
        CLRB    SUPRES              ; Init. "0" depress flag.
;+
;   QUICK PROCESS FOR "SP" CONVERSIONS.....
;-
        TST     R2                  ; Test for DP....
        BNE     50$                 ;   and branch for DP conversion...
        MOV     R3,R0               ; Else convert LO order only...
        DECB    DPFLAG              ; Set DP flag for 1 word conversion....
        BR      60$                 ;   and process...
;+
;   PREPARE HI ORDER WORD FOR ASCII CONVERSION...
;-
50$:    ASL     R2                  ; Shift HI word to enable ascii conversion.
        BIT     #100000,R3          ; MSB in LO word ??
        BEQ     70$                 ;   NO - branch...
        INC     R2                  ;  YES - add #1 to HI word..
        BIC     #100000,R3          ;          and clear the sign bit.
70$:    MOV     R2,R0               ; Input octal character is in R3.
;+
;   CONVERT FIRST WORD INTO ASCII...
;-
60$:    MOV     #PV1,R1             ; R1 --> Place value.
        MOV     #6,R2               ; R2 --> Max. no. of characters to convert.
;
NEXT:   MOV     #-1,R4              ; Initialize the digit counter....
COUNT:  INC     R4                  ; R4 is the digit counter...
        SUB     (R1),R0             ; Get multiple count...
        BHIS    COUNT               ;   branch for more...
        ADD     (R1)+,R0            ; Restore oversubtracted input..
        TSTB    SUPRES              ; Have we reached first number ??
        BLT     80$                 ;   YES -  convert ascii...
;+
;   NOTE:  Suppress leading zeros for terminal output...
;-
        TSTB    R4                  ; Test for 1st number...
        BEQ     90$                 ;   If zero then move in a blank..
        DECB    SUPRES              ;   Else no more zero supress and
        BR      80$                 ;     convert data.
90$:    MOVB    #' ,(R5)+           ; Move in blanks till we get 1st character.
        SOB     R2,NEXT             ; Branch for remaining data...
```

```
;   Conversion after leading zeros....
;-
80$:    ADD     #60,R4          ; Convert digit counter.
        MOVB    R4,(R5)+        ;   and store for output
        SOB     R2,NEXT         ; Branch for remaining data...
;+
;   FOR DP OPERATIONS:  CONVERT LOW ORDER WORD INTO ASCII NEXT...
;-
        DECB    DPFLAG          ; Dec. word count flag...
        BEQ     TYPE            ;   and type data when done...
        MOV     R3,R0           ; Else convert LO order number...
        MOV     #PV2,R1         ; R1 --> Place value (ignore MSB conversion).
        MOV     #5,R2           ;           and convert 5 ascii numbers...
        BR      NEXT            ; Branch for next conversion....
;
TYPE:   MOV     (SP)+,R4        ; RESREG....
        MOV     (SP)+,R1
        MOV     (SP)+,R0
        RTS     PC
;
        .PSECT  DATA
PV1:    .WORD   100000,10000,1000,100,10,1
PV2:    .WORD   10000,1000,100,10,1
        .EVEN
DPFLAG: .BLKB   2
SUPRES: .BLKB   0
WARN:   .ASCIZ  " ERR............RE-ENTER CHARACTER     : "<200>
;
        .TITLE  ADCOMM
;*******************************************************************;
;*  SUBROUTINE:  ADCOMM                                              *;
;*  QUERY USER FOR DATA SAMPLING PARAMETERS.  (CLKRAT,SAMRAT,NCHAN,NSCAN) *;
;*******************************************************************;
        .GLOBL  ADCOMM
        .PSECT  PROG
ADCOMM: MOV     R0,-(SP)        ; SAVREG...
        MOV     R2,-(SP)
        MOV     R3,-(SP)
;
        .PRINT  #GETCLK         ; Ask for number of CHANNELS from the user...
        JSR     PC,GETDEC       ;   Get the data from the keyboard,
        MOV     R0,CLKRAT       ;     and store in NCHAN.
;
        .PRINT  #GETSAM         ; Ask for number of SAMPLES from the user...
        JSR     PC,GETDEC       ;   Get the data from the keyboard,
        MOV     R0,SAMRAT       ;     and store in NCHAN.
;
        .PRINT  #GETCHN         ; Ask for number of CHANNELS from the user...
        JSR     PC,GETDEC       ;   Get the data from the keyboard,
        MOV     R0,NCHAN        ;     and store in NCHAN.
```

```
;   ALLOW FOR DP NO. WHEN KENNEDY TAPE IS USED.
;-
        .IF  EQ  TSAMP
          .PRINT  #GETSCN         ; Ask for a number of SCANS from the user...
          JSR     PC,GETDEC       ;   translate into machine language
          MOV     R0,NSCAN        ;     and store.
        .ENDC
;
        .IF  NE  TSAMP
          .PRINT  #GETSCN         ; Ask for a number of SCANS from the user...
          JSR     PC,GETOCT       ;   Get the DP octal number...
          MOV     R2,NSCAN        ;     and store HI order..
          MOV     R3,NSCAN+2      ;       and LO order.
        .ENDC
;
        MOV     (SP)+,R3        ; RESREG....
        MOV     (SP)+,R2
        MOV     (SP)+,R0
        RTS     PC              ;   and return...
;**********************************************************************;
        .IF  NE  XSAMP          ; START:  EXTENDED MEMORY CONDITIONALS.
        .TITLE  VIRWIN
;**********************************************************************;
;*  SUBROUTINE:  VIRWIN                                                *;
;*  SET UP OUTPUT BUFFER AS WINDOW INTO EXTENDED MEMORY.               *;
;*     Virtual Address:  40000 - 57776   (APR=2 as defined in .WDBBK)  *;
;*     Physical Memory Start Address: 100000.                          *;
;*     DYNAMIC REMAPPING TO HIGHER LOCATIONS AS DATA ACCUMULATES.      *;
;**********************************************************************;
        .GLOBL  VIRWIN
VIRWIN: MOV     R1,-(SP)        ; SAVREG...
        .CRRG   #XAREA,#XMBUF   ;DEFINE EXTENDED MEMOR  .ECION
        BCC     1$              ;NO PROBLEM
        JMP     XERROR          ;XM ERROR

1$:     MOV     XMBUF+R.GID,WINDOW+W.NRID       ;ASSOCIATE WINDOW WITH REGION
        .CRAW   #XAREA,#WINDOW              ;CREATE,MAP WINDOW TO 1st PART REGION
        BCC     2$                          ;NO P[illegible]PC
  ;   return.
;
```

```
;*****************************************************************************;
;*   SUBROUTINE:  VIRWRT                                                     *;
;*   WRITE DATA IN VIRTUAL MEMORY TO TH[redacted],R4        ;COMPUTE NUMBER
OF WINDOWS USED (MINUS 1)
        MOV     R4,WNDCNT
        MOV     #AREA,R5        ;SET UP OUTPUT
;+
;   SET #SCANS & ERC START TIME IN HEADER BLOCK.
;-
        MOV     OUTBUF,R3       ; R3 --> base address of header block.
        TST     (R3)+    [redacted]
                                        ;   WRITE DATA BUFFER TO OUTPUT FILE...
;-
        MOV     BLKNUM,Ri         ; R1 --> BLK NO.
        MOV     OUTBUF,R3         ; R3 --> Base of output buffer.
[redacted] ; Another complete block to write ??
        BLO     LASTBL              ;   NO - write out partial block.
        DEC     WNDCNT              ;   YES-[redacted]OT LAST WINDOW
        CMP     R3,OUTPTR       ; ALL DATA TRANSFERED TO DISK?
        BLO     RIGHT           ; BRANCH IF MORE
[redacted]
SAMPLE: DEC BASRAT              ;TIME TO SAMPLE?
        BEQ DOSAMP
        RTI                     ;NO - RETURN
DOSAMP: MOV BASRAT+2,BASRAT     ;YES - RESET COUNTER
        INCB SAMPFL
        RTI     ;CANNOT ACCESS EXTENDED MEMORY IN INTERRUPT HANDLER
;
        .TITLE  ERCINT
;*******[redacted] READ ERC CLOCK AND RETURN
;
        .TITLE  XERROR
;*****************************************************************************;
;*  SUBROUTINE:  XERROR                                                      *;
;*  EXTENDED MEMORY ERROR HANDLING.                                          *;
;*****************************************************************************;
        .GLOBL  XERROR
XERROR: MOVB    @ERRWRD,R0    [redacted]          .ENDC                    ; END:
 EXTENDED MEMORY CONDITIONALS.
        .IF  NE  FSAMP          ; START:   FORGROUND/BACKGROUND CONDITIONALS.
;
        .TI[redacted]             MOV     R0,R1           ;SAVE IT
        .GVAL   #CAREA,#374     ;GET RT-11 USER AREA SIZE
        SUB     R0,R1           ;ALLOCATE AREA FOR USER
        CMP     R1[redacted]     *;
;*****************************************************************************;
        .GLOBL  WRTDAT
WRTDAT: MOV #AREA,R5            ;SET UP OUTPUT
        MOV OUTBUF,R3
        CLR R4                  ;CHANNEL NUMBER
[redacted]********************************************************;
        .ENDC                   ; END:     FORGROUND/BACKGROUND CONDITIONALS.
        .IF  NE  RSAMP          ; START:   I/O DEVICE CONDITIONALS.
```

```
type dy:xmsamp.v22
!hco9

        .TITLE  XMSAMP
        .IDENT  /V2.2/
;***************************************************************************;
;*                           WESTON OBSERVATORY                            *;
;*                      DEPT. OF GEOLOGY & GEOPHYSICS                      *;
;*               GEOPHYSICAL DATA ACQUISITION SYSTEM SOFTWARE              *;
;* FILE: XMSAMP.MAC      (RT11XM USE ONLY)                                 *;
;*                                                                         *;
;* REVISED  24-JAN-86  C.J.CENTER  Install "XSAMP" as the module "XMSAMP"  *;
;*          27-feb-86  c.j.center  Rewritten for Preston A/D... (see note) *;
;*          02-mar-86  c.j.center  Increase delay "TIMER" to prevent       *;
;*                                 incomplete scan count.                  *;
;*          07-mar-86  c.j.center  Install 1.8 msec delay between DATI and *;
;*                                 DATO mode for A/D acknowledge of RSO-RUN.*;
;*          03-jun-86              Update version 2.1 to version 2.2       *;
;*          20-jun-86              Install D/A calibrations.               *;
;*                                                                         *;
;* Note:  Program data storage is base on the DRVBAR.  This register       *;
;*        contains the current data output location and is automatically  *;
;*        incremented with each strobe of data.                            *;
;***************************************************************************;
        .MCALL     .PRINT,.WRITW,.MAP
        .INCLUDE   \GDAS1I.INC\
        .INCLUDE   \GDAS2I.INC\
        .GLOBL     DKFLG,IVOLTS
        .INCLUDE   \GDAS4I.INC\
;+
;   SAMPLE PROGRAM SPECIFIC DEFINITIONS...
;-
BASE   = 172410           ; DRV11B hardware base address: 772410
DRVWCR = BASE             ; Word Count Register.
DRVBAR = BASE+2           ; Bus Address Register.
DRVCSR = BASE+4           ; Control/Status Register.
DRVVEC = 124              ; Interrupt Vector location.
T.100  = 35.              ; 100usec "SOB" time delay counter.
T.70   = 20.              ; T.70 = (100usec - "previous instructions").
T.50   = 17.              ; 50 usec time.
;   Misc...
        .PSECT  DATA
WARN:   .BYTE   'W,200  ; Define sample underflow WARNING character.
TABLE:  .WORD   177777  ; Store PRESTON RESET in a DMA table.
        .WORD   300     ; Store PRESTON RSO-RUN in next location.
;+
;   START OF SAMPLE INTERRUPT HANDLER...
;-
        .PSECT  PROG
XMSAMP: SAVREG                    ; Save registers on stack.
;+
;   ASSIGN REGISTER POINTERS TO DRV11B and SET PRIORITY...
;-

        MOV     #DRVWCR,R1        ; R1 --> DRV11B Word Count Register.
        MOV     #DRVBAR,R2        ; R2 --> DRV11B Bus Address Register.
        MOV     #DRVCSR,R3        ; R3 --> DRV11B Control Status Register.
;+
;    RESET A/D FIFO MEMORY AND PREPARE TO ACQUIRE DATA....
;-
        CLR     @R1             ; Assure WCR is zero (RDY is HI) !!!
        MOV     #-2,@R1         ; Store number of words to strobe into A/D.
        MOV     #TABLE,@R2      ; BAR --> start of table.
        BIS     #6,@R3          ; Set FUNCT1, FUNCT2 HI.
        BIC     #4,@R3          ; Set FUNCT2 LOW.
        MOV     #T.100,R0       ; Issue 100+ usec. delay after
        SOB     R0,.            ;   last FUNCT set.
        BIS     #403,@R3        ; Initiate DMA transfers to the Preston A/D.
TST2:   TST     @R1             ; DRVWCR will increment to zero when all
        BNE     TST2            ;   previous values have been strobed in.
;+
;    D/A CALIBRATION.
;-
        TSTB    DKCAL           ; Will we calibrate ??
        BEQ     10$             ;   NO - jump to continue.
        DEC     DKFLG           ;  YES - decrement D/A input voltage counter
        BNE     10$             ;         and continue till it's voltage time
        MOV     IVOLTS,@#DAC1   ;  Enter 5.0 volts on DAC1.
;
        TSTB    PULSE           ; Entering a pulse ??
        BEQ     10$             ;   NO - then constant voltage...
        MOV     #T.50,R0        ;  YES - then sent pulse low
        SOB     R0,.            ;         after 100 usec.
        CLR     @#DAC1          ; Clear DAC.
;+
;    SIT AND WAIT...
;    Wait:  1.7 msec for RSO-RUN acknowledge
;           0.2 msec for a 16. channel conversion
;           0.0125 msec for extra channel time delay.
;-
10$:    MOV     #800.,R0        ; Assert a time delay using the
        SOB     R0,.            ;   SOB instruction (2.85 usec/instr).
;+
;    PRESTON A/D ---> MEMORY TRANSFER.
;-
        MOV     NCHAN,@R1       ; Load DRVWCR with the 2's complement
        NEG     @R1             ;   No. of channels to strobe in.
        MOV     #SAMBUF,@R2     ; DMA data to the sample buffer (SAMBUF).
        BIC     #2,@R3          ; Set FUNCT1 LO for DATO transfer..
        BIS     #1,@R3          ; Start DMA transfer.
;+
;    FIRST SCAN:: RECORD DATA ACQUISITION START TIME....
;-
        TST     SCANS           ; Is this the first scan ??
        BNE     OKSKP           ;   NO - don't acknowledge ERC time.
        MOV     #ERCMSB,R0      ;POINT TO LAST ERC READOUT
        MOV     TIMPTR,R1       ;POINT TO START TIME IN FILE HEADER
        MOV     @R0,GOTIME      ;STORE ERC TIME OF TRIGGER
        MOV     (R0)+,(R1)+
        MOV     @R0,GOTIME+2
        MOV     @R0,(R1)+
;+
;   STORE SCAN HEADER INFORMATION....
;-
OKSKP:  MOV     OUTPTR,R5       ; SAVE DATA - GET OUTPUT BUFFER POINTER
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; version 2.2 update.
;       MOV     #SCANID,R4      ; ADDRESS OF START OF SCAN INDICATOR
;       MOV     (R4)+,(R5)+     ; Store the 3 byte
;       MOV     @R4,(R5)+       ;   scan header code..
;       MOV     R5,R4           ; SAVE ADDR, bitmap to store no. channels.
;       MOV     CHMAP1,(R5)+    ; Store BITMAP of CHANNELS IN SCAN
;       MOV     CHMAP2,@R5
;       BIC     #100000,(R5)+   ; CLEAR OUT ERC CHANNEL'S BIT
;       MOV     R5,-(SP)        ; SAVE DATA STORAGE START ADDRESS
;
        MOV     SCANID,(R5)+    ; Store the scan header flag.
        CLR     (R5)+           ; Clear ERC channel's bit....
        MOV     R5,R4           ;   and store current ptr. value.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;+
;   STORE SAMPLES...
;-
        MOV     #T.70,R0        ; Assure a 100+ usec. elapse after..
        SOB     R0,.            ;   DATO FUNCT setting before accessing data.
        MOV     #SAMBUF,R0      ; ADDRESS OF SAMPLE BUFFER
100$:   CMP     R0,@R2          ; Exceeded Preston's current DMA address ???
        BHIS    110$            ;   YES - exit...
        MOV     (R0)+,(R5)+     ;   NO  - move sample to output buffer and
        BR      100$            ;     get next sample...
110$:   MOV     R5,OUTPTR       ; Update OUTPUT BUFFER POINTER....
;+
;   DETERMINE NO. OF SAMPLES ACQUIRED....
;-
        MOV     R5,R0           ;CURRENT OUTPUT BUFFER POINTER
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;       SUB     (SP)+,R0        ; MINUS POINTER AT START OF DATA MOVE
        SUB     R4,R0           ;   minus pointer at start of data move.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
        ASR     R0              ;MAKE IT A WORD COUNT (divide by 2).
        MOVB    R0,-(R4)        ;STORE IT IN SCAN HEADER
        CMP     R0,NCHAN        ; Did we get all the samples ???
        BEQ     120$            ;  YES - continue
        .PRINT  #WARN           ;  NO  - print message and continue...
;+
;   SAMPLING COMPLETED ???
;-
120$:   INC     SCANS           ;COUNT SCANS
        CMP     SCANS,NSCAN     ;DONE?
        BEQ     19$             ;BRANCH WHEN DONE
;+
;   END OF WINDOW ???
;-
        CMP     R5,OUTEND           ;OVERFLOWED OUTPUT BUFFER?
        BLOS    20$                 ;BRANCH WHEN UNDER
        DEC     WNDCNT              ;CHECK NUMBER OF WINDOW REMAPPINGS
        BLE     19$                 ;NO MORE ALLOWED
;+
;   AT END OF WINDOW FILL BUFFER WITH ZEROS AND GET A NEW PAGE...
;-
        MOV     OUTEND,R0           ; Get end of window pointer and
        ADD     #64,R0              ;   adjust it to the "actual" 4k boundary.
21$:    CMP     R5,R0               ; Fill memory locations
        BPL     22$                 ;   till we reach the 4k boundary.
        CLR     (R5)+               ; Use a "zero fill"
        BR      21$                 ;   and branch till we're at the end....

22$:    ADD     #128.,WINDOU+W.NOFF     ;INCREASE MAPPING OFFSET BY 4K
        .MAP    #XAREA,#WINDOU          ;REMAP WINDOW
        BCC     23$                     ;NO PROBLEM
        JMP     XERROR                  ;XM ERROR
23$:    MOV     WINDOU+W.NBAS,OUTPTR    ;Create a new pointer...
        BR      20$                     ;  and continue sampling...
;+
;   EXIT IF ALL SCANS ACQUIRED...
;-
19$:    NEGB    TRIGON              ;FLAG FOR DONE AND/OR OVERFLOW
        CLR     #WNDCNT
20$:    RESREG                      ; Restore registers
        RTS     R5                  ;   and return.
        .END    ; XMSAMP
```

```
type dy:timer.mac
!hco9

        .TITLE TIMER
        .GLOBL TIMER,IPOKE,IPEEK
;*****************************************************************************************;
;* FILE: TIMER.MAC                                                                       *;
;* CALL: CALL TIMER(ITIME)                                                               *;
;*                                                                                       *;
;*      ITIME IS RETURNED IN AN INTEGER ARRAY OF DIMENSION 4.                            *;
;*      ARRAY ELEMENTS CONTAIN THE FOLLOWING:                                            *;
;*      ITIME(1):     DAY                                                                *;
;*           (2):     HOUR                                                               *;
;*           (3):     MINUTES                                                            *;
;*           (4):     SECONDS                                                            *;
;*                                                                                       *;
;*      Revision                 06-aug-81                                               *;
;*            THIS IS A MODIFICATION OF A PROGRAM NAMED GETIME APPEARING                 *;
;*            IN THE 8 OCT 80 EDITION OF THE AFGL SDAS: A FUNCTIONAL                     *;
;*            DESCRIPTION.                                                               *;
;*      Revision   c.j.center  05-mar-86  Renamed CLOCK.MAC to TIMER.MAC                 *;
;*****************************************************************************************;
        .INCLUDE        /GDAS4I.INC/
        .PSECT  PROG
;+
; TIMER SUBROUTINE.
;-
TIMER:  SAVREG
        MOV R5,-(SP)                ; Save fortran arguments on the stack.
        MOV #ALIST1,R5              ;SETUP FOR SR IPOKE
        JSR PC,IPOKE                ;SET CSR FOR HOURS + DAYS
        MOV #ALIST2,R5              ;SETUP FOR SR IPEEK
        JSR PC,IPEEK                ;READ HOURS + DAYS
        MOV R0,HRDAY                ;PUT RESULTS IN BUFFER
        MOV #ALIST3,R5              ;SETUP FOR SR IPOKE
        JSR PC,IPOKE                ;SET CSR FOR SEC + MIN
        MOV #ALIST2,R5              ;SETUP FOR SR IPEEK
        JSR PC,IPEEK                ;READ SEC + MIN
        MOV R0,SECMIN               ;PUT RESULTS IN BUFFER
        MOV #ALIST4,R5              ;SETUP FOR IPOKE
        JSR PC,IPOKE
        MOV #HRDAY,R4               ;ADRS OF TIME BUFFERS
;
        MOV (SP)+,R5                ;RESTORE R5
        TST (R5)+                   ;R5 NOW = START OF ARG LIST
        MOV @R5,R5                  ;ADRS OF ARRAY OF RETURNED TIME
        MOV #TABLE,R1
2$:     MOVB (R4)+,-(SP)            ;GET WORD OF RAW TIME
        MOVB (R4)+,1(SP)
        MOV (SP)+,R3
3$:     CLR R2                      ;CLEAR HIGH SHIFT REGISTER
        MOVB (R1)+,R0               ;SHIFT COUNT
        ASHC R0,R2                  ;SHIFT BITS FOR THIS COUNT INTO R2
        MOV R2,-(SP)            ;SAVE ON STACK
        CMP @SP,#10.            ;MAKE SURE NO STUCK BITS
        BLT 33$
        SUB #10.,@SP
33$:    TSTB @R1                ;MORE DIGITS IN THIS TIME UNIT?
        BGT 3$                  ;YES, THEN BRANCH
        MOV #SUM,R0
        MOVB (R1)+,R2           ;MAX POWER OF TEN NEEDED (NEGATED)
        MOV (SP)+,(R0)+         ;STORE LEAST SIGNIFICANT DIGIT
        MOV R3,(R0)+            ;STORE REMAINED OF TIME WORD
4$:     MOV (SP)+,R3            ;NEXT SIGNIFICANT BIT
        MUL (R0)+,R3            ;TIMES TENS POWER
        ADD R3,SUM              ;ADD TO SUM
        INC R2                  ;DONE ALL DIGITS FOR THIS UNIT?
        BNE 4$                  ;NO, THEN BRANCH BACK
        MOV SUM,(R5)+           ;RETURN UNIT TO ARRAY
        MOV SUM+2,R3            ;GET REMAINDER OF TIME WORD
        TSTB @R1                ;DONE WITH THIS WORD?
        BGT 3$                  ;NO, THEN BRANCH
        TSTB (R1)+              ;DONE BOTH INPUT WORDS OF TIME?
        BLT 2$                  ;NO, THEN BRANCH
5$:     RESREG
        RTS PC                  ;YES, THEN RETURN
;+
; DATA DEFINITIONS.
;-
        .PSECT  DATA
SUM:    .BLKW 2
TENS:   .WORD 10.,100.
TABLE:  .BYTE 2,4,4,-2,2,4,-1,-7,4,4,-1,4,4,-1,0
        .EVEN
HRDAY:  .BLKW                   ;HOURS AND DAYS BUFFER
SECMIN: .BLKW                   ;SECONDS AND MINUTES BUFFER
ALIST1: .WORD 2,ERCSR,THREE
ERCSR:  .WORD 167770            ;ADRS OF ERC CONTROL STATUS REG
THREE:  .WORD 3                 ;MASK FOR READING HR + DAY
ALIST2: .WORD 1,ERCOUT
ERCOUT: .WORD 167774            ;ADRS OF ERC OUTPUT REG
ALIST3: .WORD 2,ERCSR,TWO
TWO:    .WORD 2
ALIST4: .WORD 2,ERCSR,ZERO
ZERO:   .WORD 0
        .END            ;TIMER
```

END

2-81

DTIC